SIEMENS

Avant-propos, Sommaire

Première partie :
Utilisation de l'éditeur CONT

Deuxième partie :
Description du langage

Annexes

Glossaire, Index

SIMATIC S7

Langage CONT pour SIMATIC S7-300/400 Programmation de blocs

Manuel

C79000-G7077-C504-02

Informations relatives à la sécurité

Ce manuel donne des consignes que vous devez respecter pour votre propre sécurité ainsi que pour éviter des dommages matériels. Elles sont mises en évidence par un triangle d'avertissement et sont présentées, selon le risque encouru, de la façon suivante :



Danger

signifie que la non–application des mesures de sécurité appropriées **conduit** à la mort, à des lésions corporelles graves ou à un dommage matériel important.



Attention

signifie que la non-application des mesures de sécurité appropriées peut conduire à la mort, à des lésions corporelles graves ou à un dommage matériel important.



Avertissement

signifie que la non-application des mesures de sécurité appropriées peut conduire à des lésions corporelles légères ou à un dommage matériel.

Nota

doit vous rendre tout particulièrement attentif à des informations importantes sur le produit, aux manipulations à effectuer avec le produit ou à la partie de la documentation correspondante.

Utilisation conforme aux dispositions

Tenez compte des points suivants :



Attention

L'équipement ne doit être utilisée que pour les applications spécifiées dans le catalogue ou dans la description technique, et exclusivement avec des périphériques et composants recommandés par Siemens.

Marques

SIMATIC® et SINEC® sont des marques déposées par SIEMENS AG.

Les autres désignations figurant dans ce document peuvent être des marques dont l'utilisation par des tiers à leurs propres fins peut enfeindre les droits des propriétaires desdites marques.

Copyright © Siemens AG 1996 Tous droits réservés

Toute communication ou reproduction de ce support d'information, toute exploitation ou communication de son contenu sont interdites, sauf autorisation expresse. Tout manquement à cette règle est illicite et expose son auteur au versement de dommages et intérêts. Tous nos droits sont réservés, notamment pour le cas de la délivrance d'un brevet ou celui de l'enregistrement d'un modèle d'utilité.

Exclusion de responsabilité

Nous avons vérifié la conformité du contenu du présent manuel avec le matériel et le logiciel qui y sont décrits. Or des divergences n'étant pas exclues, nous ne pouvons pas nous porter garants pour la conformité intégrale. Si l'usage de ce manuel devait révéler des erreurs, nous en tiendrons compte et apporterons les corrections nécessaires dès la prochaine édition. Veuillez nous faire part de vos suggestions .

Siemens AG
Division Automatisation
Système d'automatisation industrielle (AUT 1)
Postfach 4848, D-90327 Nürnberg

© Siemens AG 1996
Nous nous réservons le droit de modifier les caractéristiques techniques.

Siemens Aktiengesellschaft
C79000-G7077-C504

Avant-propos

Objet du manuel

Ce manuel vous assiste pour la création de programmes utilisateur en langage CONT en vous expliquant les principes à suivre lors de l'écriture des programmes. Consultez l'aide en ligne pour obtenir tous les détails sur les étapes de traitement.

Ce manuel contient, en outre, une partie de référence qui décrit la syntaxe et le fonctionnement des éléments du langage de programmation CONT.

Groupe cible

Ce manuel s'adresse aux programmeurs de programmes S7, aux personnes responsables de la mise en exploitation et au personnel de maintenance. Des connaissances générales dans le domaine des techniques de l'automatisation sont requises.

Champ d'application

Ce manuel vaut pour la version 3.0 du logiciel de programmation STEP 7 ».

Norme

CONT correspond au langage « schéma à contacts » défini dans la norme CEI 1131-3. Vous trouverez plus de détails à ce sujet dans la table de correspondance à la norme dans le fichier NORM_TBL.WRI (anglais) ou NORM_TAB.WRI (allemand) de STEP 7.

Vue d'ensemble de la documentation utilisateur S7-300/400

Il existe une importante documentation utilisateur destinée à vous aider pour la configuration et la programmation d'un automate programmable S7 et dont vous vous servirez de manière sélective. Les explications et la figure ci-après doivent faciliter l'utilisation de cette documentation.

#

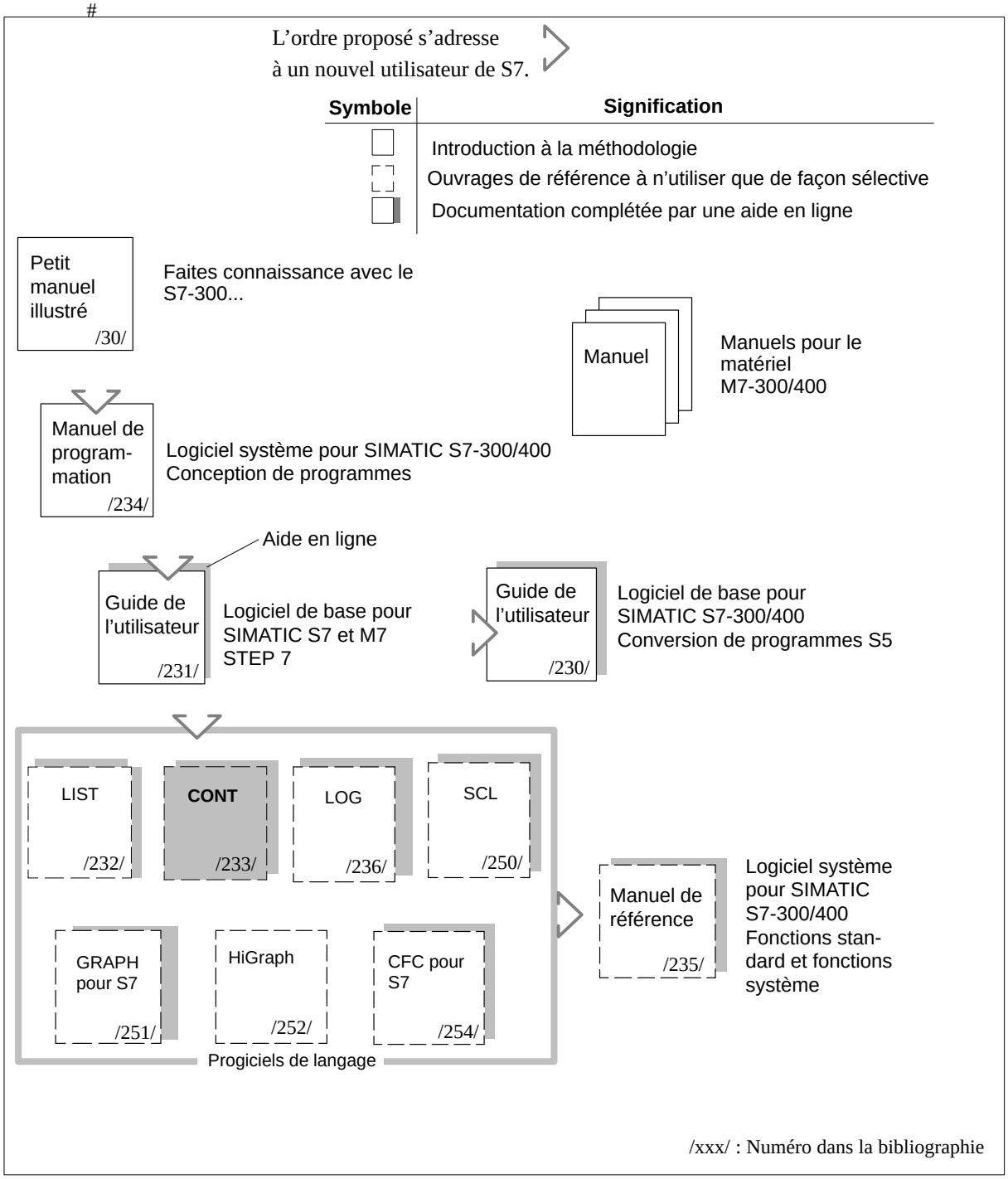

L'ordre proposé s'adresse à un nouvel utilisateur de S7.
Symbole
Signification
Introduction à la méthodologie
Ouvrages de référence à n'utiliser que de façon sélective
Documentation complétée par une aide en ligne
Petit manuel illustré /30/
Faites connaissance avec le S7-300...
Manuel
Manuels pour le matériel M7-300/400
Manuel de programmation /234/
Logiciel système pour SIMATIC S7-300/400 Conception de programmes
Aide en ligne
Guide de l'utilisateur /231/
Logiciel de base pour SIMATIC S7 et M7 STEP 7
Guide de l'utilisateur /230/
Logiciel de base pour SIMATIC S7-300/400 Conversion de programmes S5
LIST /232/
CONT /233/
LOG /236/
SCL /250/
GRAPH pour S7 /251/
HiGraph /252/
CFC pour S7 /254/
Progiciels de langage
Manuel de référence /235/
Logiciel système pour SIMATIC S7-300/400 Fonctions standard et fonctions système
/xxx/ : Numéro dans la bibliographie

Tableau 1-1 Contenu des manuels S7

Titre	Contenu
Petit manuel illustré Faites connaissance avec le S7-300	Ce manuel constitue une introduction très simple à la méthodologie de configuration et de programmation d'un automate S7-300. Il s'adresse tout particulièrement aux utilisateurs ne connaissant pas les automates programmables S7.
Manuel de programmation Conception de programmes S7-300/400	Ce manuel de programmation présente les connaissances de base sur l'organisation du système d'exploitation et d'un programme utilisateur d'une CPU S7. Il est conseillé aux nouveaux utilisateurs des S7-300/400 de l'utiliser pour avoir une vue d'ensemble de la méthodologie de programmation et pour concevoir, ensuite, leur programme utilisateur.
Manuel de référence Fonctions standard et fonctions système S7-300/400	Les CPU S7 disposent de blocs d'organisation et de fonctions système intégrés au système d'exploitation dont vous pouvez vous servir lors de la programmation. Ce manuel présente une vue d'ensemble des fonctions système, blocs d'organisation et fonctions standard chargeables disponibles dans S7, ainsi que – comme informations de référence – des descriptions d'interface détaillées pour leur utilisation dans le programme utilisateur.
Guide de l'utilisateur STEP 7	Ce guide de l'utilisateur STEP 7 explique le principe d'utilisation et les fonctions du logiciel d'automatisation STEP 7. Que vous soyez un utilisateur débutant de STEP 7 ou que vous connaissiez bien STEP 5, il vous donne une vue d'ensemble sur la marche à suivre pour la configuration, la programmation et la mise en œuvre d'un automate S7-300/S7-400. Vous pouvez, lors de l'utilisation du logiciel, accéder de manière sélective à l'aide en ligne qui répondra à vos questions précises sur le logiciel.
Guide de l'utilisateur Conversion de programmes S5	Vous aurez besoin de ce guide si vous avez l'intention de convertir des programmes S5 existants afin de les exécuter dans des CPU S7. Ce guide vous donne une vue d'ensemble du mode de fonctionnement et de l'utilisation du convertisseur ; vous trouverez des informations détaillées sur l'utilisation des fonctions du convertisseur dans l'aide en ligne. Cette dernière contient également la description d'interface des fonctions S7 converties disponibles.
Manuels LIST, LOG, CONT, SCL[1]	Les manuels concernant les progiciels de langage LIST, LOG, CONT et SCL contiennent aussi bien des instructions pour l'utilisateur que la description du langage. Vous n'avez besoin, pour la programmation d'un S7-300/400, que de l'un de ces langages, mais pouvez les mélanger à l'intérieur d'un projet si besoin est. Il est conseillé, lors de la première utilisation des langages de se familiariser avec la méthodologie de la création de programmes à l'aide du manuel. Dans le logiciel, vous pouvez appeler l'aide en ligne qui répondra à vos questions détaillées sur l'utilisation des éditeurs et compilateurs associés.
Manuels GRAPH[1] , HiGraph[1], CFC[1]	Les langages GRAPH, HiGraph et CFC offrent des possibilités supplémentaires pour la réalisation de commandes séquentielles, de graphes d'état ou de câblages graphiques de blocs. Ces manuels contiennent aussi bien des instructions pour l'utilisateur que la description du langage. Il est conseillé, lors de la première utilisation de ces langages, de se familiariser avec la méthodologie de la création de programmes à l'aide du manuel. Dans le logiciel, vous pouvez appeler l'aide en ligne (excepté pour HiGraph) qui répondra à vos questions détaillées sur l'utilisation des éditeurs et compilateurs associés.

1 Logiciels optionnels pour le logiciel système des S7-300/400

Structure du manuel

Dans ce manuel pour CONT, nous supposons que vous avez acquis les connaissances théoriques sur les programmes S7, présentées dans le manuel de programmation **/234/**. Comme les progiciels de langage se fondent sur le logiciel de base STEP 7, il est recommandé de savoir se servir de ce logiciel de base présenté dans le guide de l'utilisateur **/231/**.

Ce manuel est organisé selon les thèmes suivants :

- La première partie explique comment utiliser l'éditeur.
- La deuxième partie est conçue comme une section de référence pour toutes les opérations CONT.
- Le glossaire contient la définition des concepts les plus importants.
- L'index vous permet de retrouver rapidement les passages relatifs à des thèmes précis.

Conventions

Les renvois à d'autres publications sont indiqués à l'aide de numéros entre barres obliques /.../. Vous trouverez, à l'aide de ces numéros, le titre exact de ces publications dans la bibliographie à la fin du manuel.

Aide supplémentaire

Adressez-vous à votre agence Siemens pour toute question sur le logiciel décrit à laquelle vous ne trouveriez pas de réponse dans la documentation papier ou dans l'aide en ligne. Vous trouverez les adresses des agences et représentations Siemens dans l'annexe des publications **/70/** et /**100**/ ou dans des catalogues et dans Compuserve (go autforum). Vous pouvez bien sûr aussi appeler notre ligne directe :

Tél. + 49 (911) 895-7000 (télécopie 7001).

Si vous avez des questions ou des remarques sur le présent manuel, nous vous prions de compléter le formulaire à la fin du manuel et de l'envoyer à l'adresse indiquée. N'hésitez pas à également indiquer votre appréciation personnelle du manuel.

Nous proposons des cours pour faciliter l'apprentissage des automates programmables SIMATIC S7. Adressez-vous à votre centre de formation ou à notre centre principal à :

D-90327 Nürnberg, tél. (49) 911 / 895 3154.

Remarque

La première partie de ce manuel ne contient pas d'instructions de travail précises décomposées en étapes détaillées, car notre but est de vous présenter les procédures de principe. Vous trouverez, dans l'aide en ligne, des informations plus précises sur les différentes boîtes de dialogue du logiciel et leur traitement.

Sommaire

Avant-propos ..... **iii**

Première partie : Utilisation de l'éditeur CONT

1 Présentation du produit **1-1**

2 Introduction **2-1**

2.1 Structure d'un programme utilisateur 2-2

2.2 Création d'un programme utilisateur, aperçu 2-4

2.3 Règles à observer 2-7

3 Création de blocs de code **3-1**

3.1 Création de blocs de code, aperçu 3-2

3.2 Blocs de code dans l'éditeur 3-3

3.3 Table de déclaration des variables 3-6

3.4 Edition des tables de déclaration des variables 3-8

3.5 Déclaration de multi-instances 3-10

3.6 Affectation d'attributs système à des paramètres 3-11

3.7 Edition de la section d'instructions, aperçu 3-13

3.8 Règles fondamentales de saisie des éléments CONT 3-15

3.9 Saisie des éléments CONT 3-18

3.10 Création de branches parallèles 3-21

3.11 Edition d'adresses et de paramètres 3-23

3.12 Adressage symbolique 3-24

3.13 Modification en mode de substitution 3-26

3.14 Saisie de titres et de commentaires 3-28

4 Création de blocs de données et de types de données utilisateur **4-1**

4.1 Création de blocs de données, aperçu 4-2

4.2 Choix de la méthode de création 4-4

4.3 Edition de la table de déclaration 4-5

4.4 Edition des valeurs de données en cours 4-6

4.5 Création de types de données utilisateur 4-8

5 Edition des propriétés de bloc et test du programme 5-1

5.1 Edition des propriétés de bloc 5-2

5.2 Test du programme CONT, aperçu 5-5

5.3 Paramétrage de l'affichage pour visualisation d'état de programme 5-6

5.4 Paramétrage des conditions de déclenchement 5-7

5.5 Sélection d'environnement de test et lancement ou arrêt de la visualisation d'état de programme 5-8

Deuxième partie : Description du langage

6 Structure et éléments de CONT 6-1

6.1 Eléments et pavés 6-2

6.2 Logique booléenne et tables de vérité 6-6

6.3 Signification des registres de la CPU dans les instructions 6-12

7 Adressage 7-1

7.1 Présentation 7-2

7.2 Types d'opérandes 7-4

8 Opérations combinatoires sur bits 8-1

8.1 Présentation 8-2

8.2 Contact à fermeture 8-3

8.3 Contact à ouverture 8-4

8.4 Sortie 8-5

8.5 Connecteur 8-6

8.6 Inverser RLG 8-7

8.7 Sauvegarder RLG dans RB 8-8

8.8 Mettre à 1 8-9

8.9 Mettre à 0 8-10

8.10 Initaliser compteur 8-11

8.11 Incrémenter 8-12

8.12 Décrémenter 8-13

8.13 Temporisation sous forme d'impulsion 8-14

8.14 Temporisation sous forme d'impulsion prolongée 8-15

8.15 Temporisation sous forme de retard à la montée 8-16

8.16 Temporisation sous forme de retard à la montée mémorisé 8-17

8.17 Temporisation sous forme de retard à la retombée 8-18

8.18 Détecter front montant du RLG 8-19

8.19 Détecter front descendant du RLG 8-20

8.20 Détecter front montant de signal 8-21

8.21 Détecter front descendant de signal 8-22

8.22 Bascule mise à 1, mise à 0 8-23

8.23 Bascule mise à 0, mise à 1 8-24

9 Opérations de temporisation 9-1
9.1 Adresse d'une temporisation en mémoire et composants d'une temporisation .. 9-2
9.2 Choix de la temporisation correcte 9-4
9.3 Temporisation sous forme d'impulsion 9-5
9.4 Temporisation sous forme d'impulsion prolongée 9-7
9.5 Temporisation sous forme de retard à la montée 9-9
9.6 Temporisation sous forme de retard à la montée mémorisé 9-11
9.7 Temporisation sous forme de retard à la retombée 9-13
10 Opérations de comptage 10-1
10.1 Adresse d'un compteur en mémoire et composants d'un compteur 10-2
10.2 Compteur incrémental/décrémental 10-3
10.3 Compteur incrémental 10-5
10.4 Compteur décrémental 10-7
11 Opérations arithmétiques sur nombres entiers 11-1
11.1 Additionner entiers de 16 bits 11-2
11.2 Additionner entiers de 32 bits 11-3
11.3 Soustraire entiers de 16 bits 11-4
11.4 Soustraire entiers de 32 bits 11-5
11.5 Multiplier entiers de 16 bits 11-6
11.6 Multiplier entiers de 32 bits 11-7
11.7 Diviser entiers de 16 bits 11-8
11.8 Diviser entiers de 32 bits 11-9
11.9 Reste de division (32 bits) 11-10
11.10 Evaluation des bits du mot d'état pour les opérations arithmétiques sur nombres entiers 11-11
12 Opérations arithmétiques sur nombres réels 12-1
12.1 Présentation 12-2
12.2 Additionner nombres réels 12-3
12.3 Soustraire nombres réels 12-4
12.4 Multiplier nombres réels 12-5
12.5 Diviser nombres réels 12-6
12.6 Evaluation des bits du mot d'état pour les opérations arithmétiques sur nombres réels 12-7
12.7 Valeur absolue d'un nombre réel 12-8
12.8 Carré ou racine carrée d'un nombre réel 12-9
12.9 Logarithme naturel d'un nombre réel 12-11
12.10 Valeur exponentielle d'un nombre réel 12-12
12.11 Fonctions trigonométriques d'angles sous forme de nombres réels 12-13

13 Opérations de comparaison 13-1
13.1 Comparer entiers de 16 bits 13-2
13.2 Comparer entiers de 32 bits 13-3
13.3 Comparer nombres réels 13-5
14 Opérations de transfert et de conversion 14-1
14.1 Affecter valeur 14-2
14.2 Convertir nombre DCB en entier de 16 bits 14-4
14.3 Convertir entier de 16 bits en nombre DCB 14-5
14.4 Convertir entier de 16 bits en entier de 32 bits 14-6
14.5 Convertir nombre DCB en entier de 32 bits 14-7
14.6 Convertir entier de 32 bits en nombre DCB 14-8
14.7 Convertir entier de 32 bits en nombre réel 14-9
14.8 Complément à 1 d'entier de 16 bits 14-10
14.9 Complément à 1 d'entier de 32 bits 14-11
14.10 Complément à 2 d'entier de 16 bits 14-12
14.11 Complément à 2 d'entier de 32 bits 14-13
14.12 Inverser le signe d'un nombre réel 14-14
14.13 Arrondir à entier de 32 bits 14-15
14.14 Tronquer à la partie entière (32 bits) 14-16
14.15 Convertir nombre réel en entier supérieur le plus proche 14-17
14.16 Convertir nombre réel en entier inférieur le plus proche 14-18
15 Opérations combinatoires sur mots 15-1
15.1 Présentation 15-2
15.2 ET mot 15-3
15.3 ET double mot 15-4
15.4 OU mot 15-5
15.5 OU double mot 15-6
15.6 OU exclusif mot 15-7
15.7 OU exclusif double mot 15-8
16 Opérations de décalage et de rotation 16-1
16.1 Opérations de décalage 16-2
16.2 Opérations de rotation 16-10
17 Opérations sur blocs de données 17-1
17.1 Ouvrir bloc de données 17-2

18 Opérations de saut 18-1
18.1 Présentation 18-2
18.2 Saut si 1 (inconditionnel) 18-3
18.3 Saut si 1 (conditionnel) 18-4
18.4 Saut si 0 18-5
18.5 Repère de saut 18-6
19 Opérations sur bits d'état 19-1
19.1 Présentation 19-2
19.2 Bit d'anomalie « Registre RB » 19-3
19.3 Bits de résultat 19-4
19.4 Bit d'anomalie « Opération illicite » 19-6
19.5 Bit d'anomalie « Débordement » 19-7
19.6 Bit d'anomalie « Débordement mémorisé » 19-9
20 Opérations de gestion d'exécution de programme 20-1
20.1 Appeler FC/SFC sans paramètre 20-2
20.2 Appeler FB, FC, SFB, SFC et multi-instances 20-4
20.3 Retour 20-7
20.4 Opérations du relais de masquage 20-8
20.5 Activer/désactiver relais de masquage 20-9
20.6 Relais de masquage en fonction/hors fonction 20-12

Annexes

A Liste alphabétique des opérations A-1
A.1 Liste des désignations françaises A-2
A.2 Liste des désignations françaises et des désignations internationales (anglaises) correspondantes A-5
A.3 Liste des désignations internationales (anglaises) A-9
A.4 Liste des désignations internationales (anglaises) et des désignations françaises correspondantes A-12
B Exemples de programmation B-1
B.1 Présentation B-2
B.2 Opérations combinatoires sur bits B-3
B.3 Opérations de temporisation B-7
B.4 Opérations de comptage et de comparaison B-11
B.5 Opérations arithmétiques sur nombres entiers B-13
B.6 Opérations combinatoires sur mots B-14
C Représentation des nombres C-1
C.1 Représentation des nombres C-2

D Bibliographie **D-1**
Glossaire **Glossaire-1**
Index **Index-1**

Première partie : Utilisation de l’éditeur CONT

Présentation du produit 1

Introduction 2

Création de blocs de code 3

Création de blocs de données et de types de données utilisateur 4

Edition des propriétés de bloc et test du programme 5

Présentation du produit

1

Que signifie CONT

Le schéma à contacts CONT est un langage de programmation graphique. La syntaxe de ses instructions s'inspire des schémas à relais : CONT permet de suivre facilement le flux d'énergie circulant via des entrées, des sorties et des opérations entre les barres d'alimentation.

Langage de programmation CONT

Le langage de programmation CONT met à votre disposition tous les éléments nécessaires à la création d'un programme utilisateur complet. Il dispose d'un jeu d'opérations très important. Vous disposez de diverses opérations de base différentes ainsi que d'une large palette d'opérandes et d'adressages. Cela vaut également pour le concept des fonctions et des blocs fonctionnels qui vous permettent de structurer clairement un programme CONT.

Logiciel de programmation

Le logiciel de programmation CONT est une partie intégrante du logiciel de base STEP 7. Après l'installation du logiciel STEP 7, vous disposez ainsi de toutes les fonctions d'édition, de compilation et de test pour CONT.

CONT vous permet de créer votre programme utilisateur en utilisant un éditeur incrémental. La structure de données locale peut être saisie de façon très conviviale par l'intermédiaire d'éditeurs de tables.

En plus du langage CONT, les langages de programmation LIST et LOG sont également intégrés dans le logiciel de base. Vous pouvez donc passer d'un langage à l'autre en sélectionnant la représentation adéquate pour la programmation d'un bloc.

De manière générale, les programmes écrits en CONT et en LOG peuvent être représentés sans problème en LIST. Lors de la conversion de programmes CONT en programmes LOG, et vice versa, tout élément de programme ne pouvant être représenté dans le langage cible sera représenté en LIST.

Introduction

2

Présentation

Ce chapitre contient une description abrégée de la structure d'un programme utilisateur composé de blocs.

L'éditeur CONT se base sur le SIMATIC Manager (gestionnaire de projets SIMATIC), le fondement de toutes les applications STEP 7. Ce chapitre décrit le passage du SIMATIC Manager à l'éditeur CONT ainsi que la place tenue par les blocs créés dans la structure de projet.

Contenu de ce chapitre

Paragraphe	Thème	Page
2.1	Structure d'un programme utilisateur	2-2
2.2	Création d'un programme utilisateur, aperçu	2-4
2.3	Règles à observer	2-7

2.1 Structure d’un programme utilisateur

Blocs de code et blocs de données

Un programme utilisateur est composé de blocs de code et de blocs de données. On appelle blocs de code tous les blocs contenant une section d’instructions, c’est-à-dire les blocs d’organisation, les blocs fonctionnels et les fonctions.

Blocs d’organisation

Les blocs d’organisation (OB) représentent l’interface entre le système d’exploitation et le programme utilisateur. Une tâche précise incombe à chaque bloc d’organisation. Vous pouvez composer le programme utilisateur CONT de votre CPU S7 des blocs d’organisation (OB) dont vous avez besoin pour votre solution d’automatisation. Dans le cas le plus simple, il s’agit des blocs d’organisation destinés :

- à la mise en route (OB100, OB101),
- au programme principal cyclique (OB1) et
- au traitement des erreurs (OB80 à OB87, OB121, OB122), dans le cas où une erreur ne doit pas entraîner l’arrêt de votre CPU.

Il existe d’autres blocs d’organisation vous permettant de traiter des alarmes de la CPU ou du processus.

Pour savoir quelle tâche est affectée à quel bloc d’organisation et quelle information de déclenchement est fournie par le système d’exploitation de la CPU, veuillez consulter le manuel de référence **/235/**.

Fonctions et blocs fonctionnels

Vous pouvez programmer chaque bloc d’organisation en tant que programme structuré en créant des fonctions (FC) et des blocs fonctionnels (FB) et en appelant ceux-ci dans la section d’instructions. Lors de l’appel de bloc, vous devez fournir des données correspondant aux paramètres déclarés.

- Un bloc fonctionnel (FB) est un bloc de code avec rémanence. Dans ce cas, la mémoire est un bloc de données d’instance affecté au bloc fonctionnel, dans lequel sont sauvegardés les paramètres effectifs et les données statiques du bloc fonctionnel.
- Une fonction (FC) est un bloc de code sans rémanence. Après le traitement des fonctions, les paramètres de sortie contiennent les valeurs de fonction calculées. C’est ensuite à vous d’organiser l’utilisation et la sauvegarde des paramètres effectifs selon vos besoins.

Données

Le système d’exploitation met à votre disposition les données suivantes :

- Entrées et sorties de périphérie
- Mémoire image des entrées et des sorties
- Mémentos
- Temporisations
- Compteurs

Vous pouvez, en outre, définir vos propres données :

- Les données globales valables pour l'ensemble du programme utilisateur peuvent être définies dans les blocs de données.
- Les variables statiques sont uniquement valables dans le bloc fonctionnel dans lequel elles sont définies. A chaque appel de bloc fonctionnel, vous précisez un bloc de données d'instance contenant les données statiques en plus de tous les paramètres. Si des multi-instances sont définies, leurs données d'instance sont insérées avec leurs données statiques dans le bloc de données d'instance.
- Les données temporaires sont définies lors de la création de blocs de code. Elles occupent uniquement de l'espace mémoire dans la pile pendant le traitement du bloc de code.

Blocs de données

Les blocs de données mémorisent les données du programme utilisateur. On distingue les blocs de données globaux et les blocs de données d'instance.

- Vous pouvez accéder aux blocs de données globaux à partir de tout endroit du programme utilisateur.
- Les blocs de données d'instance sont affectés à un bloc fonctionnel et contiennent, en plus des données du bloc fonctionnel, aussi les données de multi-instances éventuellement définies. Aussi est-il conseillé d'accéder au bloc de données d'instance uniquement en relation avec ce bloc fonctionnel.

Informations supplémentaires

Le manuel de programmation **/234/** contient une introduction à la méthodologie de la programmation.

2.2 Création d'un programme utilisateur, aperçu

Programme utilisateur

Un programme utilisateur devant être exécuté dans une CPU S7 est essentiellement constitué de blocs. Il contient, en outre, des informations supplémentaires, telles que les données destinées à la configuration ou à la mise en réseau du système. En fonction de l'application, vous devez donc créer les blocs suivants pour votre programme utilisateur :

- blocs d'organisation (OB),
- blocs fonctionnels (FB),
- fonctions (FC),
- blocs de données (DB).

Pour simplifier la programmation, vous pouvez, en outre, définir vous-même des types de données (UDT) que vous pouvez alors utiliser comme types de données propres ou comme modèles pour la création de blocs de données.

Les blocs souvent utilisés tels que les blocs fonctionnels système (SFB) et les fonctions système (SFC) sont intégrés dans la CPU. D'autres blocs (les blocs destinés aux fonctions CEI ou les blocs de régulation, par exemple) peuvent être chargés sous forme de progiciel (blocs fonctionnels standard). Vous pouvez intégrer très simplement dans votre programme utilisateur ces blocs que vous ne programmez pas vous-même.

Nota

Pour l'affichage en ligne des blocs fonctionnels système et des fonctions système intégrés dans votre CPU, sélectionnez **Système cible ▸ Etat du module**.

Editeur CONT incrémental

Le logiciel de base STEP 7 comprend un éditeur destiné à la programmation de blocs. Pour programmer des blocs de code (blocs d'organisation, blocs fonctionnels et fonctions), vous pouvez sélectionner le langage CONT. L'éditeur CONT est un éditeur incrémental, c'est-à-dire qu'il procède à une vérification de la syntaxe après chaque entrée que vous effectuez. Les erreurs de syntaxe sont affichées et les opérations illicites immédiatement refusées.

Lancement à partir du SIMATIC Manager

L'éditeur CONT est lancé à partir du SIMATIC Manager, à condition d'y avoir créé auparavant un projet avec un programme S7. Vous pouvez créer le programme en fonction ou indépendamment du matériel. A cet effet, insérez le programme S7 directement sous le projet ou éditez le programme S7 affecté au module programmable. Le programme peut comprendre, entre autres, des classeurs pour le programme utilisateur (blocs), des sources ou des diagrammes.

L'éditeur CONT est uniquement destiné à l'édition de blocs mémorisés dans le classeur pour le programme utilisateur.

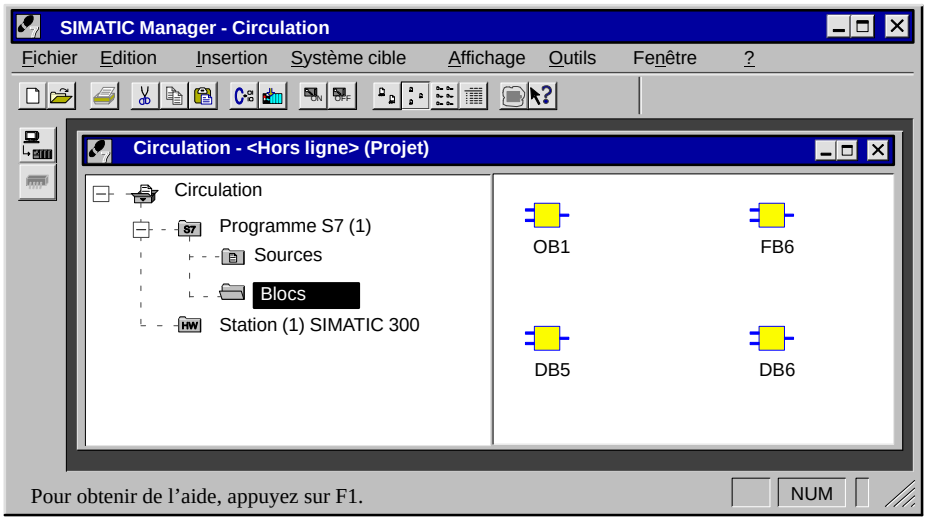


Figure 2-1 Lancement de l'éditeur CONT à partir du SIMATIC Manager

Création d'un bloc

Lorsque vous voulez créer un nouveau bloc pour la première fois, vous devez d'abord créer un bloc vide dans le SIMATIC Manager via lequel vous appelez l'éditeur. Une fois que vous avez appelé l'éditeur CONT, vous pouvez y créer d'autres blocs.

- Sélectionnez le classeur « Blocs » dans le SIMATIC Manager et insérez un bloc du type souhaité en sélectionnant **Insérer** ▸ **Bloc S7** ▸ **....** Le nouveau bloc apparaît dans le côté droit de la fenêtre de projet.
- Dans l'éditeur, vous pouvez créer un nouveau bloc par l'intermédiaire de la commande **Fichier** ▸ **Nouveau**. Dans la boîte de dialogue qui apparaît alors, indiquez le type et le numéro de bloc.

Définition du langage de programmation

Lors de la création du bloc, vous définissez le langage de programmation que vous souhaitez utiliser. L'éditeur correspondant est alors lancé au moment de l'ouverture du bloc. Pour programmer en langage CONT, sélectionnez le langage de création « CONT ».

Ouverture d'un bloc

Pour ouvrir un bloc, cliquez deux fois sur celui-ci dans le SIMATIC Manager. Vous pouvez également ouvrir un bloc en sélectionnant **Edition** ▸ **Ouvrir un objet** ou à l'aide de l'icône correspondante dans la barre d'outils.

Sauvegarde et chargement d'un bloc

Lors de la sauvegarde d'un bloc dans l'éditeur, vous devez tenir compte des points suivants :

- La commande **Fichier** ▸ **Enregistrer** sauvegarde toujours le bloc dans le classeur « Blocs » sur le disque dur de la console de programmation ou du PC.
- La commande **Système cible** ▸ **Charger** charge le bloc ouvert dans la CPU.

Après avoir créé les blocs pour votre programme utilisateur, chargez le programme intégral du SIMATIC Manager dans la CPU S7. Pour toute information supplémentaire concernant le chargement de programmes utilisateur, veuillez consulter le manuel utilisateur **/231/**.

Nota

Il n'est pas toujours suffisant de charger individuellement dans la CPU tous les blocs que vous avez créés, car des données de la configuration système peuvent également être requises. Il est donc toujours conseillé de charger tout d'abord l'intégralité du programme par l'intermédiaire du SIMATIC Manager.

Appel de fonctions d'assistance

Dans l'éditeur, vous pouvez exécuter les fonctions suivantes destinées à fournir une aide supplémentaire lors de la création et de la mise en service d'un programme :

Tableau 2-1 Fonctions d'assistance dans l'éditeur CONT

Fonctions	Commande
Appel des données de référence de programmes utilisateur	**Outils ▸ Données de référence**
Edition de la table des mnémoniques ou de mnémoniques individuels	**Outils ▸ Table des mnémoniques** ou **Outils ▸ Editer les mnémoniques**
Visualisation/forçage de variables	**Système cible ▸ Visualiser/forcer des variables**
Affichage/modification de l'état de fonctionnement et effacement général de la CPU	**Système cible ▸ Etat de fonctionnement** ou **Système cible ▸ Effacement général**
Affichage de l'état des modules de la CPU	**Système cible ▸ Etat du module**
Modification de la date et de l'heure de la CPU	**Système cible ▸ Mettre à l'heure**

Une description complète de ces fonctions se trouve dans le manuel utilisateur **/231/**.

2.3 Règles à observer

Séquence de création des blocs

Lors de la création de blocs de code et de blocs de données d'un programme utilisateur, la séquence de création des blocs est d'une importance décisive. Veuillez toujours respecter la règle suivante : **les blocs appelés à partir d'un autre bloc doivent déjà exister lors de la programmation de leur appel**, c'est-à-dire avoir été programmés auparavant. Il est impossible de saisir un bloc inexistant comme élément CONT (pavé). Si vous appelez un bloc inexistant à l'aide de la fonction CALL, un message d'erreur est affiché lors de la sauvegarde, puisque le bloc appelé n'existe pas dans le programme.

Edition pendant l'exécution du programme

STEP 7 permet d'éditer en ligne le programme utilisateur chargé dans une CPU pendant que la CPU se trouve à l'état de fonctionnement « Marche » (RUN).



Attention

Le fait de modifier un programme en ligne lorsqu'il est en cours d'exécution peut entraîner des erreurs de fonctionnement ou des perturbations imprévues dans l'installation et donc des blessures ou des dommages matériels.

Si la CPU fonctionne en ligne et se trouve à l'état de marche (RUN) lors de l'édition du programme utilisateur chargé dans la CPU, vous risquez de provoquer des situations dans lesquelles les machines sont subitement mises en marche ou arrêtées et donc d'être à l'origine de blessures corporelles ou de dommages matériels.

Concevez toujours le déroulement du processus conformément aux règles de sécurité en vigueur. N'essayez jamais de modifier un programme en ligne en cours d'exécution sans avoir pris des mesures de sécurité supplémentaires.

Nota

Pour de plus amples informations sur la commutation en ligne et hors ligne, veuillez consulter le manuel utilisateur **/231/**.

Création de blocs de code

3

Présentation

Un programme utilisateur ne peut pas être exécuté sans blocs de code. Pour la résolution de nombreux problèmes, vous pouvez certes utiliser les blocs intégrés de l'unité centrale ou les blocs fonctionnels standard disponibles dans le système. Cependant, vous devez créer vous-même un nombre minimal de blocs de code. Ce chapitre décrit cette procédure dans l'éditeur CONT.

Contenu de ce chapitre

Paragraphe	Thème	Page
3.1	Création d'un bloc de code, aperçu	3-2
3.2	Blocs de code dans l'éditeur	3-3
3.3	Tables de déclaration des variables	3-6
3.4	Edition des tables de déclaration des variables	3-8
3.5	Déclaration de multi-instances	3-10
3.7	Edition de la section d'instructions, aperçu	3-13
3.8	Règles fondamentales de saisie des éléments CONT	3-15
3.9	Saisie des éléments CONT	3-18
3.10	Création de branches parallèles	3-21
3.11	Edition des adresses et des paramètres	3-23
3.12	Adressage symbolique	3-24
3.13	Modification en mode de substitution	3-26
3.14	Saisie de titres et de commentaires	3-28

3.1 Création de blocs de code, aperçu

Blocs de code

Les blocs de code (blocs d'organisation, blocs fonctionnels et fonctions) sont constitués d'une section de déclaration de variables et d'une section d'instructions. Ils sont, en outre, dotés de propriétés. Lors de la programmation, vous devez donc traiter les trois sections suivantes :

- La **table de déclaration des variables**. Dans la table de déclaration des variables, vous définissez les paramètres, les attributs système pour paramètres ainsi que les variables locales du bloc.
- La **section d'instructions**. Dans la section d'instructions, vous programmez le code du bloc devant être traité par l'automate programmable. Il est constitué d'un ou de plusieurs réseaux comprenant des instructions CONT.
- Les **propriétés de bloc**. Les propriétés de bloc contiennent des informations supplémentaires entrées par le système, telles que l'horodatage ou l'indication du chemin. En outre, vous pouvez indiquer vous-même le nom, la famille, la version et l'auteur (voir au chapitre 5), et affecter des attributs système aux blocs.

Edition d'un bloc de code

L'ordre dans lequel vous traitez les trois sections d'un bloc de code n'a pas d'importance. Vous pouvez évidemment corriger et compléter le bloc ultérieurement.

Si vous souhaitez accéder à des mnémoniques de la table des mnémoniques, nous vous recommandons de d'abord vérifier si celle-ci est complète et de la compléter, le cas échéant.

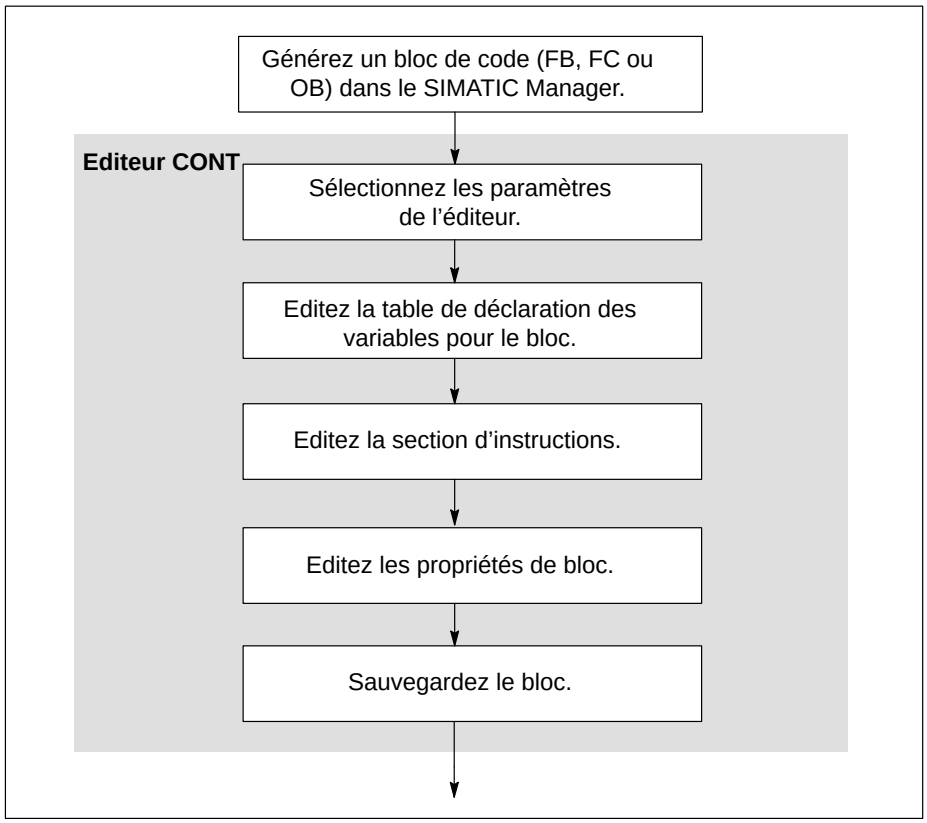

Générez un bloc de code (FB, FC ou OB) dans le SIMATIC Manager.
Editeur CONT
Sélectionnez les paramètres de l'éditeur.
Editez la table de déclaration des variables pour le bloc.
Editez la section d'instructions.
Editez les propriétés de bloc.
Sauvegardez le bloc.

Figure 3-1 Marche à suivre pour programmer des blocs de code en CONT

3.2 Blocs de code dans l'éditeur

Aperçu

Avant de commencer la programmation dans l'éditeur CONT, vous devriez vous familiariser avec les sélections possibles afin de pouvoir travailler de façon aussi conviviale que possible et conformément à vos habitudes.

Paramètres de l'éditeur

Une boîte de dialogue à onglets s'ouvre lorsque vous exécutez **Outils ▸ Paramètres**. Dans la page d'onglet « Editeur », vous pouvez définir des présélections pour la programmation de blocs, à savoir :

- la police de caractères (type, style et taille) des textes et des tables
- le langage que vous souhaitez utiliser (LOG, CONT ou LIST). En fonction du langage sélectionné, un nouveau bloc sera ouvert en LOG, en CONT ou en LIST. Quel que soit votre choix, vous pourrez afficher ultérieurement le bloc dans les langages non sélectionnés, à condition de respecter quelques restrictions.
- si pour un nouveau bloc, vous désirez que soient affichés dès l'abord les mnémoniques et les commentaires.

Vous pouvez modifier les paramètres pour le langage, les commentaires et les mnémoniques pendant l'édition à l'aide de commandes du menu **Affichage ▸ ...**.

Paramètres pour CONT

La page d'onglet « CONT/LOG » à laquelle vous accédez également via **Outils ▸ Paramètres**, vous permet de définir les paramètres de base suivants.

- Mise en page CONT : paramètre qui définit la taille de représentation de vos réseaux. En fonction de la taille, vous pouvez ainsi varier le nombre d'éléments CONT consécutifs que vous souhaitez placer dans une branche. La définition de la taille de représentation se répercute également sur l'impression de votre bloc.
- Largeur de zone d'opérande : paramètre qui définit la largeur de la zone de texte réservée aux opérandes (adresses, par exemple). En cas de dépassement de la largeur de zone, l'éditeur va à la ligne. Pour l'adressage symbolique, il est nécessaire d'utiliser une zone d'opérande de grande taille alors qu'une zone de petite taille suffit pour l'adressage absolu.
- Ligne/couleur pour : élément sélectionné,
  contacts,
  état satisfait
  état non satisfait.

Fenêtre principale de l'éditeur CONT

Après l'ouverture d'un bloc de code apparaît une fenêtre comportant :

- la **table de déclaration des variables** du bloc dans la partie supérieure
- dans la partie inférieure, la **section d'instructions** dans laquelle vous éditez le code du bloc.

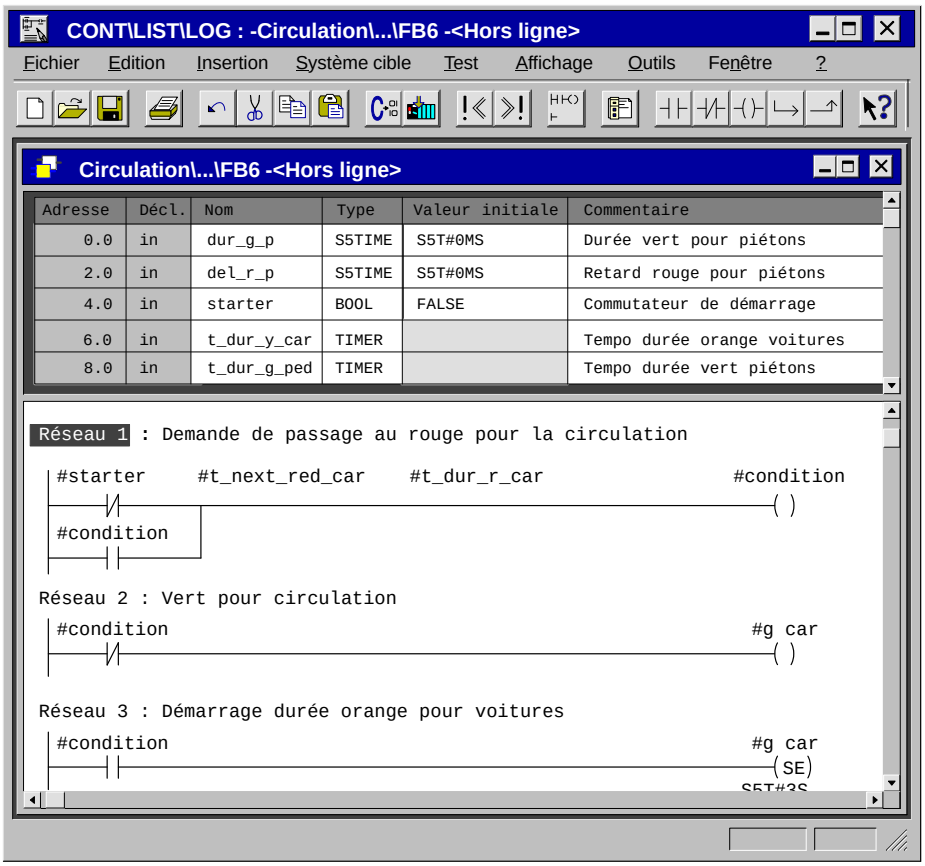

CONT\LIST\LOG : -Circulation\...\FB6 -<Hors ligne>
Fichier Edition Insertion Système cible Test Affichage Outils Fenêtre ?
Circulation\...\FB6 -<Hors ligne>
Adresse | Décl. | Nom | Type | Valeur initiale | Commentaire
0.0 | in | dur_g_p | S5TIME | S5T#0MS | Durée vert pour piétons
2.0 | in | del_r_p | S5TIME | S5T#0MS | Retard rouge pour piétons
4.0 | in | starter | BOOL | FALSE | Commutateur de démarrage
6.0 | in | t_dur_y_car | TIMER | | Tempo durée orange voitures
8.0 | in | t_dur_g_ped | TIMER | | Tempo durée vert piétons
Réseau 1 : Demande de passage au rouge pour la circulation
#starter #t_next_red_car #t_dur_r_car #condition
#condition
Réseau 2 : Vert pour circulation
#condition #g car
Réseau 3 : Démarrage durée orange pour voitures
#condition #g car
SE

Figure 3-2 Table de déclaration des variables et section d'instructions en CONT

Les **propriétés de bloc** peuvent être éditées dans une boîte de dialogue propre (cf. chapitre 5).

Vous pouvez ouvrir simultanément et traiter en alternance plusieurs blocs dans l'éditeur.

Relations entre la déclaration de variables et la section d'instructions

La table de déclaration des variables et la section d'instructions sont étroitement liées entre elles, étant donné que les noms utilisés dans la table de déclaration sont repris dans la section d'instructions. Par conséquent, toute modification effectuée dans la déclaration des variables se répercute dans la section d'instructions, sur l'ensemble du bloc.

Tableau 3-1 Relations entre la déclaration de variables et la section d'instructions

Action dans la déclaration de variables	Réaction dans la section d'instructions
Entrée correcte	S'il existait du code incorrect, une variable non déclarée auparavant devient correcte.
Modification correcte du nom sans modification du type	Le mnémonique est immédiatement représenté partout avec le nouveau nom.
Un nom correct est modifié en un nom incorrect.	Le code n'est pas modifié.
Un nom incorrect est modifié en un nom correct.	S'il existait du code incorrect, il devient correct.
Modification de type	S'il existait du code incorrect, il devient correct et inversement.
Suppression d'une variable utilisée dans un code	Le code correct devient incorrect.
Modifications de commentaire	Aucune
Saisie erronée d'une nouvelle variable	Aucune
Suppression d'une variable non utilisée	Aucune
Modification de valeur initiale	Aucune

3.3 Table de déclaration des variables

Aperçu

Les variables locales, y compris les paramètres formels du bloc et les attributs système pour paramètres, sont définies dans la table de déclaration des variables. Il en résulte que :

- lorsque vous effectuez la déclaration, vous réservez en même temps un espace mémoire correspondant dans la pile des données locales ou dans le bloc de données d'instance ;
- en définissant des paramètres d'entrée, de sortie et d'entrée/sortie, vous définissez également « l'interface » pour l'appel du bloc dans le programme ;
- lorsque vous déclarez des variables dans un FB, ces variables conditionnent en même temps la structure de données de chaque DB d'instance affecté au FB ;
- en définissant des attributs système, vous conférez des propriétés spéciales aux paramètres en ce qui concerne la configuration de messages et de liaisons, les fonctions de contrôle-commande et la configuration du système de conduite.

Structure de la table de déclaration des variables

Après avoir ouvert un bloc de code que vous avez créé, une table de déclaration des variables sélectionnée par défaut est affichée. Dans cette table figurent uniquement les types de déclaration valables pour le type de bloc sélectionné (IN, OUT, IN_OUT, STAT, TEMP) dans l'ordre prescrit.

Lors de la création d'un bloc d'organisation, une déclaration des variables standard dont vous pouvez modifier les valeurs est affichée.

La table de déclaration des variables contient l'adresse, la déclaration, le nom, le type de données, la valeur initiale et le commentaire décrivant la variable. Chaque ligne contient une déclaration de variable, sauf les variables du type Tableau (ARRAY) ou Structure (STRUCT) qui requièrent plusieurs lignes.

Circulation\...\FB40 - <Hors ligne>

Adresse	Décl.	Nom	Type	Valeur initiale	Commentaire
0.0	in	Marche	BOOL	FALSE	Allumer lumière
0.1	in	Démarr	BOOL	FALSE	Commutateur
2.0	out	Moteur	BOOL	FALSE	Moteur
2.1	out	Message	BOOL	FALSE	Moteur
4.0	in_out	Marche_sort1	INT	0	
6.0	in_out	Marche_sort2	INT	0	

Figure 3-3 Exemple de table de déclaration des variables

Signification des colonnes

Les colonnes de la table de déclaration des variables ont la signification suivante :

Tableau 3-2 Colonnes de la table de déclaration des variables

Colonne	Signification	Remarques	Traitement
Adresse	Adresse en format OCTET.BIT	Pour des données qui requièrent plus d'un octet, l'adresse indique l'affectation par un saut vers l'adresse d'octet suivante. * : taille d'un élément de tableau en octets + : adresse de début, rel. au début de structure = : espace mémoire total requis pour une structure	Entrée système : l'adresse est définie par le système et affichée au moment où vous achevez la saisie d'une déclaration.
Déclaration	Type de déclaration, « utilisation de la variable »	Entrées possibles en fonction du type de bloc : Paramètre d'entrée « in » Paramètre de sortie « out » Paramètre d'entrée/sortie « in_out » Variable statique « stat » Variable temporaire « temp »	Présélection système en fonction du type de bloc
Nom	Nom symbolique (mnémonique) de la variable	Le nom de variable commence toujours par une lettre. Les mots-clés réservés ne sont pas autorisés.	obligatoire
Type	Type de données de la variable (BOOL, INT, WORD, ARRAY, etc.)	Vous pouvez sélectionner les types de données élémentaires par le menu du bouton droit de votre souris.	obligatoire
Valeur initiale	Valeur initiale lorsque le logiciel ne doit pas reprendre la valeur par défaut.	La valeur entrée doit être compatible avec le type de données. La valeur initiale est validée comme valeur en cours de la variable lors de la première sauvegarde d'un bloc de données, à moins d'avoir défini explicitement une valeur en cours.	facultatif
Commentaire	Commentaire à des fins de documentation		facultatif

Signification du « drapeau »

L'icône « petit drapeau » apparaît dans la colonne « Nom » (cf. figure 3-3) lorsque vous avez affecté des attributs système à une variable (cf. paragraphe 3.7). Double-cliquez sur ce drapeau pour que s'affiche la boîte de dialogue « Attributs système ».

Modification de la largeur des colonnes

La largeur des colonnes peut être modifiée en fonction de vos besoins. Pour ce faire, positionnez le pointeur de la souris sur la ligne qui sépare deux colonnes. Gardez le bouton gauche de la souris enfoncé et glissez la souris horizontalement jusqu'à atteindre la largeur souhaitée. Vous pouvez également modifier la largeur des colonnes en exécutant la commande **Affichage ▸ Largeur de colonne** sur la table sélectionnée. Si vous ne souhaitez pas afficher les colonnes facultatives « Valeur initiale » et « Commentaire », vous pouvez les masquer et afficher uniquement les colonnes auxquelles vous portez le plus d'intérêt.

3.4 Edition des tables de déclaration des variables

Procédure

Lorsque vous créez une déclaration, respectez l'ordre de saisie : nom, type, valeur initiale (facultative) et commentaire (facultatif). Utilisez la touche de tabulation TAB pour déplacer le curseur dans le champ suivant. L'adresse est affectée à la variable lorsque la ligne est complétée.

Une vérification de syntaxe est effectuée à la fin de la saisie dans un champ. Les erreurs sont affichées en rouge. Il n'est pas nécessaire de corriger les erreurs tout de suite ; vous pouvez poursuivre la saisie et effectuer les corrections ultérieurement.

Fonctions d'édition

Pour éditer la table, vous disposez des fonctions du menu **Edition**. Pour une édition plus simple, vous pouvez utiliser le menu du bouton droit de votre souris qui varie en fonction du contexte.

Pour entrer le type de données, vous pouvez également appeler le menu du bouton droit de la souris. Lorsque vous sélectionnez le menu « Type de données », l'ensemble des types de données élémentaires est affiché.

Pour sélectionner des lignes individuelles, cliquez sur le champ d'adresse correspondant, protégé en écriture. Pour sélectionner des lignes supplémentaires du même type de déclaration, maintenez enfoncée la touche MAJ pendant la sélection. Les entrées sélectionnées sont présentées sur fond noir.

Modification du type de déclaration

La colonne « Décl. » est protégée en écriture. La position de la déclaration dans la table détermine son type. Ainsi, les variables ne peuvent être saisies que dans l'ordre correct conformément au type de déclaration. Lorsque vous souhaitez modifier un type de déclaration, coupez la déclaration et insérez-la ensuite sous le type de déclaration qu'elle doit adopter.

Saisie de structures

Pour saisir une structure comme variable, entrez son nom dans la colonne « Nom », STRUCT dans le champ du type de données et appuyez sur la touche TAB ou sur la touche d'entrée à la fin de la ligne. Une ligne vierge est insérée dans le tableau et une ligne contenant le mot-clé END_STRUCT pour indiquer la fin de la structure. Saisissez les éléments de structure dans les lignes vierges en définissant leur nom, leur type de données et, facultativement, leur valeur initiale. Vous pouvez insérer des éléments (lignes) supplémentaires par l'intermédiaire de la commande de menu ou en appuyant sur la touche d'entrée.

Pour sélectionner une structure entière, cliquez sur le champ « Adresse » de la première ou de la dernière ligne (donc dans la ligne contenant STRUCT ou END_STRUCT). Pour sélectionner une seule déclaration dans une structure, cliquez sur le champ « Adresse » de la ligne considérée.

Lorsque vous saisissez des structures sur plusieurs niveaux (comprises dans une autre structure), la hiérarchie est mise en évidence par l'indentation des noms de variable.

Saisie de tableaux

Pour entrer un tableau comme type de données, entrez son nom dans la colonne « Nom », le mot-clé ARRAY dans le champ du type de données en indiquant la taille du tableau, par exemple array[1..20,3..24] pour un tableau à deux dimensions. Lorsque vous appuyez sur la touche TAB (il se peut que vous deviez appuyer sur ladite touche à plusieurs reprises), une ligne est insérée dans laquelle vous entrez le type de données pour le tableau.

Pour sélectionner un tableau (array), cliquez sur le champ « Adresse » d'une ligne correspondante.

Vous pouvez définir la valeur initiale de chaque élément de tableau avec une valeur individuelle ou à l'aide d'un facteur de répétition (voir figure 3-4) :

- Valeur individuelle : vous attribuez une valeur initiale propre à chaque élément. Vous saisissez les valeurs sous forme de liste, séparées par des virgules.
- Facteur de répétition : vous attribuez une valeur initiale identique à plusieurs éléments. La valeur est entrée entre parenthèses, précédée du facteur de répétition correspondant au nombre d'éléments.

Exemple

La figure 3-4 montre l'exemple d'une table de déclaration des variables.

Circulation\...\FB50 - <Hors ligne>

Adresse	Décl.	Nom	Type	Valeur initiale	Commentaire
0.0	in	Structur1	STRUCT		
+0.0	in	var1	BOOL	FALSE	
+2.0	in	var2	INT	0	
+4.0	in	var3	WORD	W#16#0	
=6.0	in		END_STRUCT		
6.0	in	tableau1	ARRAY[1..20,1..40]	TRUE	
*2.0	in		BOOL		

Figure 3-4 Structures et tableaux dans une table de déclaration des variables

Nota

Si vous modifiez a posteriori la déclaration de variables de blocs dont vous avez déjà programmé l'appel, il pourra éventuellement y avoir des conflits d'horodatage. Par conséquent, programmez d'abord tous les blocs appelés et ensuite les blocs appelants correspondants. Pour les blocs fonctionnels, vous devez, le cas échéant, recréer les blocs de données d'instance correspondants.

Si vous modifiez a posteriori un UDT que vous avez indiqué comme type de donnée dans une déclaration de variables, vous devez vérifier la déclaration de variables du bloc et sauvegarder à nouveau le bloc.

3.5 Déclaration de multi-instances

Multi-instances

Une multi-instance est créée lors de la déclaration d'une variable statique du type de données d'un bloc fonctionnel. Dans la section d'instructions, l'instance est appelée comme élément CONT.

Pour de plus amples informations sur le rôle des multi-instances, veuillez consulter le manuel de programmation **/234/**. La syntaxe de l'appel de multi-instances est décrite dans la deuxième partie de ce manuel.

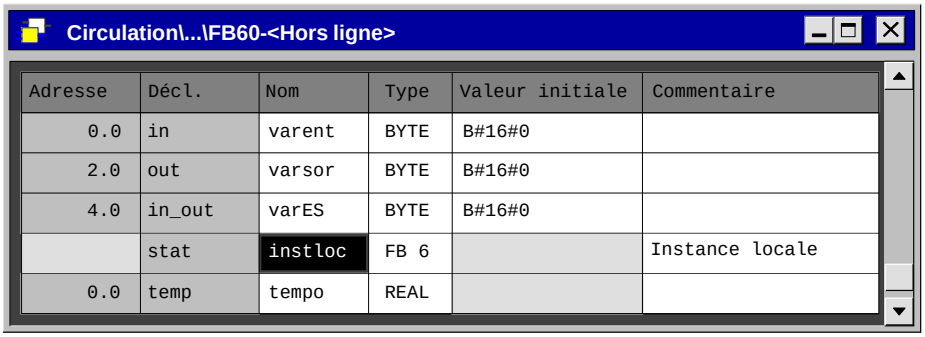
Règles

Pour la déclaration de multi-instances, veuillez respecter les règles suivantes :

- Les multi-instances peuvent uniquement être déclarées dans des blocs fonctionnels.
- Un bloc de données d'instance doit être affecté au bloc fonctionnel dans lequel une multi-instance est déclarée.
- Une multi-instance peut être déclarée uniquement comme variable statique (type de déclaration « stat »).

Saisie de multi-instances

Pour déclarer une multi-instance, entrez le nom de la variable précédé du type de déclaration « stat ». Entrez le bloc fonctionnel comme type de données. Le nom du bloc fonctionnel peut être absolu ou symbolique. L'entrée d'un commentaire est facultative.



Adresse	Décl.	Nom	Type	Valeur initiale	Commentaire
0.0	in	varent	BYTE	B#16#0	
2.0	out	varsor	BYTE	B#16#0	
4.0	in_out	varES	BYTE	B#16#0	
	stat	instloc	FB 6		Instance locale
0.0	temp	tempo	REAL		

Figure 3-5 Déclaration de multi-instances (exemple)

3.6 Affectation d'attributs système à des paramètres

Attributs système

Il est possible d'affecter des attributs système à des blocs et à des paramètres. Ces attributs gèrent la configuration de messages et de liaisons, les fonctions de contrôle-commande ainsi que la configuration du système de conduite.

Vous pouvez affecter des attributs système aux paramètres dans la table de déclaration des variables.

Saisie d'attributs système pour paramètres

Pour affecter des attributs système à un paramètre, sélectionnez le nom de ce paramètre dans la table de déclaration des variables et exécutez la commande **Edition ▸ Propriétés de l'objet**. Dans la page d'onglet « Attributs système » de la boîte de dialogue « Propriétés », entrez l'attribut désiré et la valeur correspondante.

Le tableau 3-3 présente les attributs système que vous pouvez préciser dans la table de déclaration des variables.

Tableau 3-3 Attributs système pour paramètres

Attribut	Valeur	Vous affectez cet attribut ...	Type de déclaration autorisé
S7_server	connection, alarm_archiv	lorsque le paramètre est significatif pour la configuration de messages ou de liaisons (ce paramètre contenant le numéro de liaison ou de message).	IN
S7_a_type	alarm, alarm_8, alarm_8p, alarm_s, notify, ar_send	lorsque le paramètre doit définir le type d'un bloc de signalisation (condition préalable : l'attribut S7_server:=alarm_archiv est également affecté).	IN, uniquement pour des blocs de type FB, SFB
S7_co	pbkl, pbk, ptpl, obkl, fdl, iso, pbks, obkv	lorsque le paramètre doit préciser le type de liaison lors de la configuration de liaisons (condition préalable : l'attribut S7_server:=connection est également affecté).	IN
S7_m_c	true, false	lorsque le paramètre doit être contrôlé ou commandé à partir d'un appareil de contrôle-commande.	IN/OUT/ IN_OUT, uniquement pour des blocs de type FB, SFB
S7_shortcut	2 caractères quelconques, par exemple W, Y	lorsque des symboles doivent être affectés au paramètre pour l'évaluation de valeurs analogiques.	IN/OUT/ IN_OUT, uniquement pour des blocs de type FB, SFB
S7_unit	Unité, par exemple litre	lorsque des unités doivent être affectées au paramètre pour l'évaluation de valeurs analogiques.	IN/OUT/ IN_OUT, uniquement pour des blocs de type FB, SFB

Tableau 3-3 Attributs système pour paramètres, (suite)

Attribut	Valeur	Vous affectez cet attribut ...	Type de déclaration autorisé
S7_strin_0	16 caractères quelconques, par exemple OU-VERT	lorsque du texte doit être affecté au paramètre pour l'évaluation de valeurs binaires.	IN/OUT/ IN_OUT, uniquement pour des blocs de type FB, SFB, FC, SFC
S7_strin_1	16 caractères quelconques, par exemple FERME	lorsque du texte doit être affecté au paramètre pour l'évaluation de valeurs binaires.	IN/OUT/ IN_OUT, uniquement pour des blocs de type FB, SFB, FC, SFC
S7_visible	true, false	lorsque le paramètre ne doit pas être affiché en CFC.	IN/ OUT/IN_OUT, uniquement pour des blocs de type FB, SFB, FC, SFC
S7_link	true, false	lorsque le paramètre ne doit pas pouvoir être connecté en CFC.	IN/ OUT/IN_OUT, uniquement pour des blocs de type FB, SFB, FC, SFC
S7_dynamic	true, false	lorsqu'il doit être possible de dynamiser le paramètre en CFC lors du test.	IN/ OUT/IN_OUT, uniquement pour des blocs de type FB, SFB, FC, SFC
S7_param	true, false	lorsqu'il faut protéger le paramètre d'un paramétrage incorrect en CFC.	IN/IN_OUT, uniquement pour des blocs de type FB, SFB, FC, SFC

3.7 Edition de la section d'instructions, aperçu

Section d'instructions

Vous décrivez, dans la section d'instructions, le déroulement du programme de votre bloc de code. A cet effet, vous entrez des éléments CONT dans des réseaux. La section d'instructions d'un bloc de code est constituée, dans la plupart des cas, de plusieurs réseaux. L'éditeur effectue une vérification de syntaxe immédiatement après la saisie d'un élément CONT et affiche, le cas échéant, les saisies incorrectes en rouge. Le positionnement incorrect d'éléments est également refusé avec émission d'un message d'erreur.

Eléments éditables de la section d'instructions

Vous pouvez éditer le titre du bloc, les commentaires de bloc, le titre du réseau, les commentaires de réseau et les instructions dans les réseaux d'une section d'instructions.

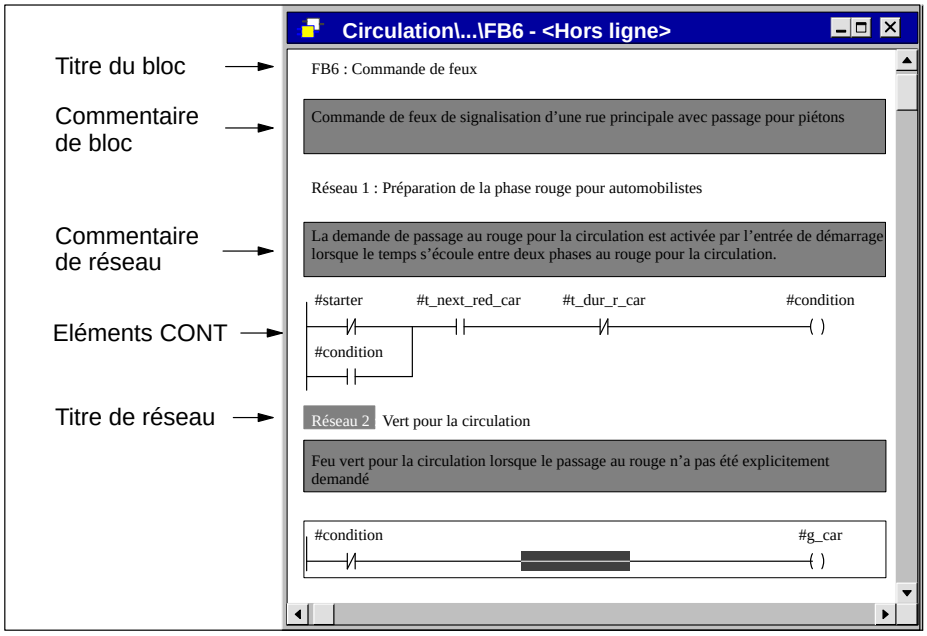

Titre du bloc
Commentaire de bloc
Commentaire de réseau
Eléments CONT
Titre de réseau
Circulation\...\FB6 - <Hors ligne>
FB6 : Commande de feux
Commande de feux de signalisation d'une rue principale avec passage pour piétons
Réseau 1 : Préparation de la phase rouge pour automobilistes
La demande de passage au rouge pour la circulation est activée par l'entrée de démarrage lorsque le temps s'écoule entre deux phases au rouge pour la circulation.
#starter
#t_next_red_car
#t_dur_r_car
#condition
#condition
Réseau 2
Vert pour la circulation
Feu vert pour la circulation lorsque le passage au rouge n'a pas été explicitement demandé
#condition
#g_car

Figure 3-6 Constitution de la section d'instructions

Saisie de nouveaux blocs

En principe, vous pouvez éditer les éléments constituants de la section d'instructions dans un ordre quelconque. Si vous programmez un bloc pour la première fois, nous vous recommandons de procéder de la manière ci-après.

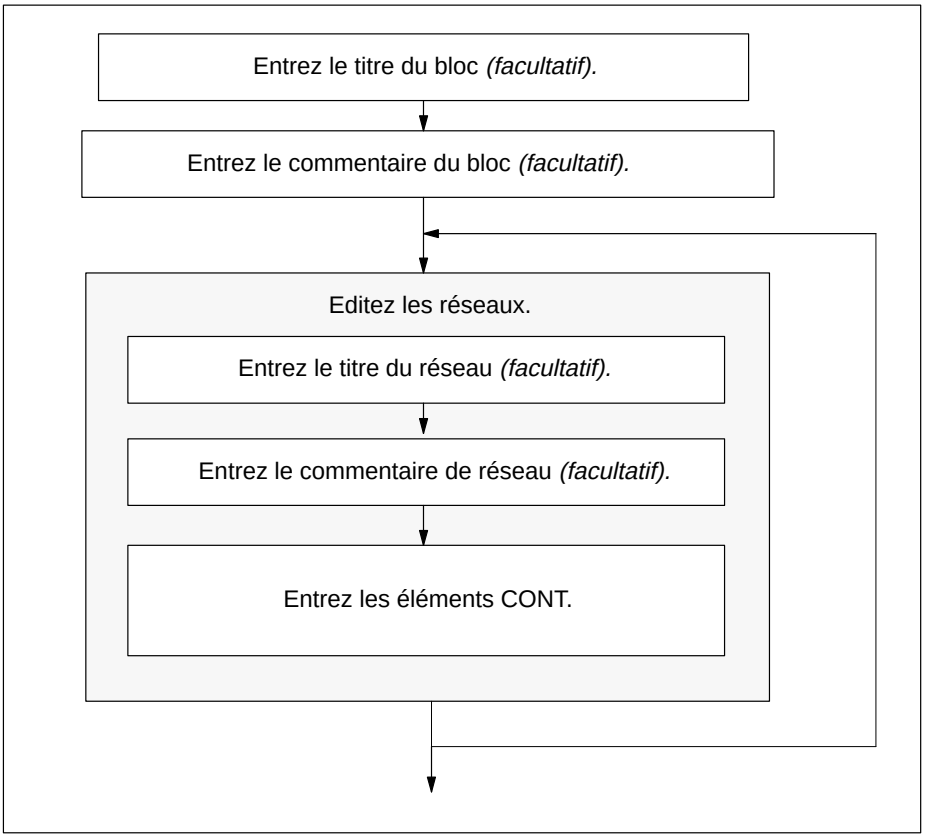

Entrez le titre du bloc (facultatif).
Entrez le commentaire du bloc (facultatif).
Editez les réseaux.
Entrez le titre du réseau (facultatif).
Entrez le commentaire de réseau (facultatif).
Entrez les éléments CONT.

Figure 3-7 Edition de la section d'instructions

Vous pouvez procéder aux modifications aussi bien en mode d'insertion qu'en mode de substitution. Pour passer d'un mode à l'autre, appuyez sur la touche d'insertion.

3.8 Règles fondamentales de saisie des éléments CONT

Généralités

Un réseau CONT peut être formé par plusieurs éléments dans plusieurs branches. Tous les éléments et branches doivent être reliés entre eux, la barre d'alimentation gauche n'étant pas considérée comme connexion (CEI 1131-3).

La programmation en CONT est soumise à quelques règles. Le système vous assiste en affichant, le cas échéant, des messages d'erreur.

Fermeture d'un réseau CONT

Chaque réseau CONT doit avoir une fermeture sous forme de bobine ou de pavé. Néanmoins les éléments CONT suivants ne peuvent en aucun cas servir de dernier élément d'un réseau :

- Pavés de comparaison
- Bobines pour connecteurs —(#)—
- Bobines pour l'évaluation des fronts montants —(P)— ou descendants —(N)—

Flux d'énergie

La saisie de deux branches qui pourraient entraîner un flux d'énergie en sens opposé (exemple à la figure 3-8) est irréalisable. Lorsque l'état du signal en E 1.4 est « 0 », le flux d'énergie en E 6.8 irait de droite à gauche, ce qui est inadmissible.

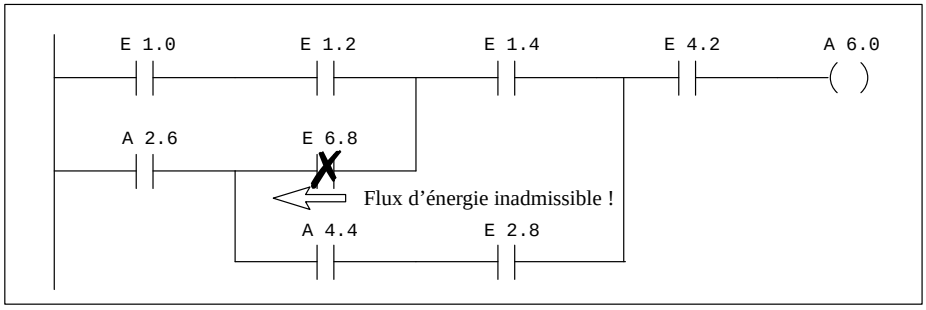


Figure 3-8 Courant électrique en sens opposé (inadmissible)

Court-circuit

Vous ne pouvez pas éditer de branches formant un court-circuit. La figure 3-9 en donne un exemple :

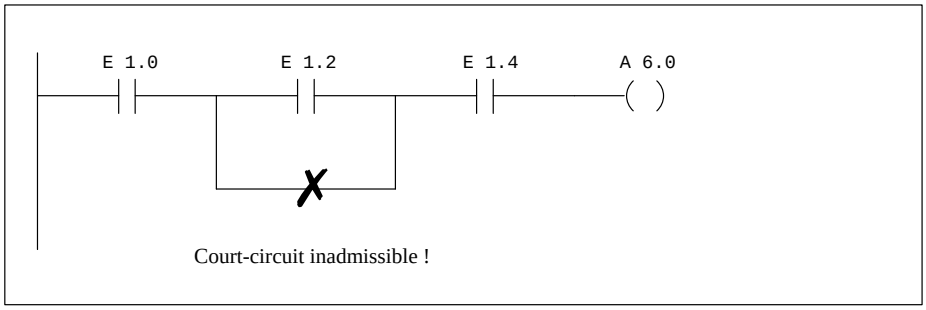


Figure 3-9 Court-circuit dans un réseau CONT (inadmissible)

Disposition des pavés

Le point de départ d'une branche contenant un pavé doit toujours être la barre d'alimentation gauche. La branche en amont du pavé peut toutefois comprendre des opérations logiques ou d'autres pavés.

Ne placez pas non plus les pavés dans une dérivation en T. Les pavés de comparaison font exception à cette règle. La figure 3-10 en donne un exemple :

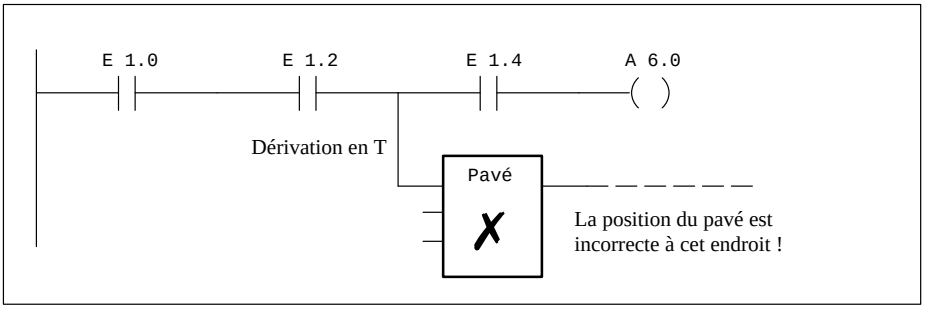


Figure 3-10 Pavé dans une dérivation en T (inadmissible)

Disposition des bobines

Les bobines sont placées automatiquement au bord droit du réseau où elles constituent la fermeture d'une branche.

Exceptions : les bobines pour connecteurs —(#)— et les bobines pour l'évaluation des fronts montants —(P)— ou descendants —(N)— ne peuvent être placées ni tout à gauche ni tout à droite de la branche. Elles ne sont pas admises non plus dans les branchements en parallèle.

Parmi les bobines, on retrouve celles qui requièrent une opération booléenne, et celles pour lesquelles les opérations booléennes sont interdites.

- **Bobines requérant des opérations booléennes :**

  Sortie —( ), Mise à 1 de la sortie —(S), Mise à 0 de la sortie—(R)

  Connecteur —(#)—, Front montant —(P)—, Front descendant —(N)—

  toutes les bobines de compteur et de temporisation

  Saut si 0 —(JMPN)

  Relais de masquage en fonction —(MCR<)

  Charger le résultat logique dans le registre RB —(SAVE)

  Retour —(RET)

- **Bobines n'autorisant pas les opérations booléennes :**

  Activer relais de masquage —(MCRA)

  Désactiver relais de masquage —(MCRD)

  Ouvrir bloc de données —(OPN)

  Relais de masquage hors fonction —(MCR>)

Toutes les autres bobines peuvent avoir une opération booléenne ou non.

L'utilisation des bobines suivantes pour une **sortie parallèle est à proscrire** :

Saut si 0 —(JMPN)

Saut si 1 —(JMP)

Appel de bloc —(CALL)

Retour —(RET)

Entrée/sortie de validation

La connexion de l'entrée de validation « EN » ou de la sortie de validation « ENO » des pavés est possible, mais n'est pas obligatoire.

Suppression et modification

La suppression d'un élément d'une branche entraîne la suppression de la branche entière si la branche ne comportait que cet élément.

La suppression d'un pavé entraîne la suppression de toutes les branches reliées au pavé par des entrées booléennes, à l'exception de la branche principale.

Le mode de substitution vous permet de remplacer commodément des éléments d'un même type (voir paragraphe 3.13).

3.9 Saisie des éléments CONT

Saisie des éléments CONT

Pour sélectionner un réseau, cliquez sur la désignation du réseau (« Réseau 1 », par exemple). Vous pouvez, par exemple, couper, à nouveau insérer ou copier le réseau ainsi sélectionné.

Pour créer un nouveau réseau, sélectionnez la commande **Insertion ▸ Réseau** ou cliquez sur l'icône correspondante dans la barre d'outils. Le nouveau réseau est inséré sous le réseau sélectionné. Il ne contient qu'une seule branche avec une bobine.

Lorsque vous saisissez plus d'éléments que votre écran ne peut en contenir, le réseau est décalé vers la gauche. Sélectionnez les commandes de menu **Affichage ▸ Agrandir/Réduire/Facteur d'agrandissement...** pour obtenir une meilleure vue d'ensemble de votre réseau.

Sélection dans les réseaux

Pour accéder à un réseau, cliquez sur un élément CONT de ce réseau. Dans un réseau, vous pouvez sélectionner de manière générale trois types de zones à l'aide de la souris :

- les éléments CONT, par exemple un contact ou un pavé,
- des points de jonction,
- des éléments vides (segment de fil ou branche ouverte).

Vous ne pouvez sélectionner qu'une de ces zones à la fois. La figure 3-11 illustre la sélection des éléments (plusieurs sélections sont représentées en même temps).

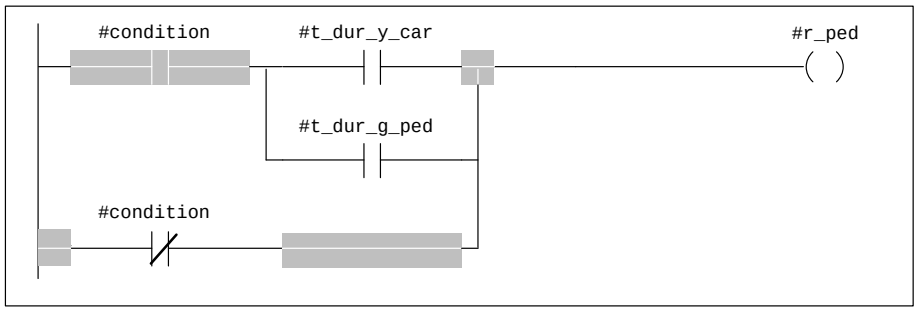


Figure 3-11 Sélections possibles dans un réseau CONT

Vous pouvez définir vous-même la couleur des sélections. A cet effet, ouvrez la page d'onglet « CONT/LOG » en sélectionnant **Outils ▸ Paramètres**.

Saisie d'éléments CONT

Vous disposez de plusieurs possibilités pour insérer des éléments CONT :

- Insérez le contact à fermeture, le contact à ouverture ou la bobine avec les touches de fonction F2, F3 ou F4.
- Sélectionnez l'icône du contact à fermeture, du contact à ouverture ou de la bobine dans la barre d'outils.
- Sélectionnez un élément dans le menu **Insertion ▸ Eléments CONT**.
- Sélectionnez la commande **Insertion ▸ Eléments de programme**, l'icône correspondante dans la barre d'outils ou appuyez sur la touche de fonction F11. Choisissez les éléments souhaités dans la boîte de sélection **Eléments de programme** qui s'affiche alors.

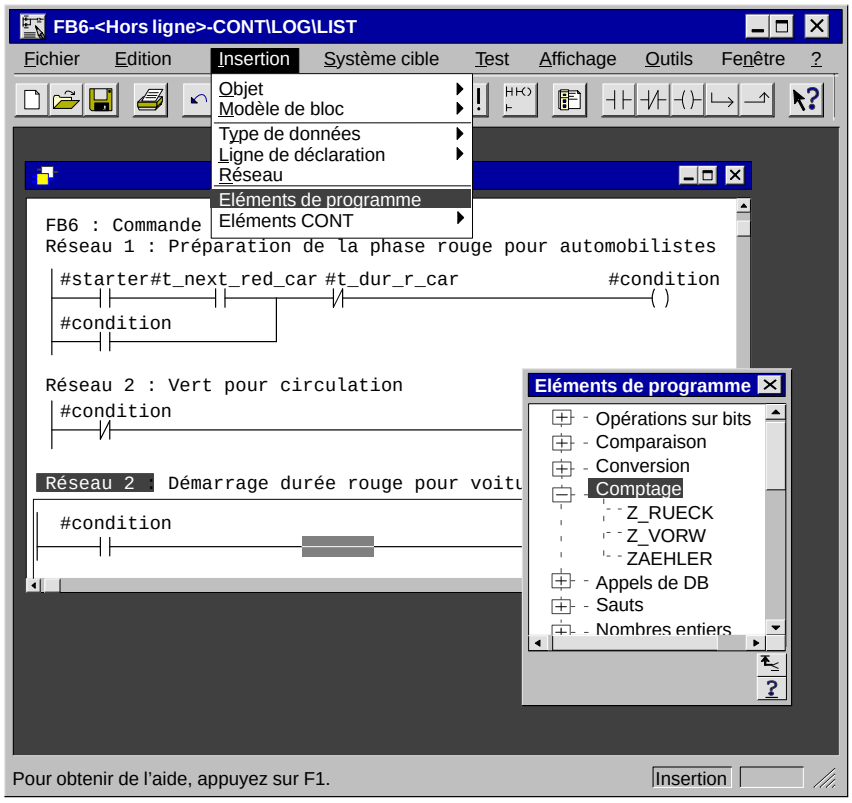


Figure 3-12 Insertion d'élément CONT via la boîte de dialogue de sélection

L'élément CONT choisi est inséré derrière l'élément sélectionné.

Nota

Si vous choisissez le groupe « Blocs FB » ou « Blocs FC », dans la boîte de sélection « Eléments de programme », la liste indique tous les blocs correspondants intervenant dans le programme CPU. Si vous choisissez le groupe « Blocs SFC » ou « Blocs SFB », tous les blocs fonctionnels système ou les fonctions système disponibles sur la CPU sont listés.

Quant à la sélection du groupe « Bibliothèques », elle entraîne l'énumération des bibliothèques standard de STEP 7 et des bibliothèques que vous avez créées vous-même.

De cette manière, vous pouvez incorporer dans votre réseau des blocs entiers et programmer en très peu de temps par exemple l'appel d'autres blocs.

Appel de multi-instances

Vous appelez les multi-instances que vous avez définies dans la table de déclaration des variables comme élément CONT. A cet effet, utilisez la commande de menu **Insertion ▸ Eléments de programme**. La liste des éléments CONT comporte un groupe « multi-instances » dans lequel figurent toutes les multi-instances déclarées.

3.10 Création de branches parallèles

Application

Pour créer des combinaisons logiques OU dans le langage de programmation CONT, vous devez créer des branches parallèles.

Règles

Pour traiter les branches parallèles, veuillez observer les règles suivantes :

- Insérez les branches OU de gauche à droite.
- Les branches parallèlles s'ouvrent vers le bas et se ferment vers le haut.
- Pour ouvrir une branche parallèle, soit sélectionnez la commande de menu **Insertion ▸ Eléments CONT ▸ Ouvrir branche,** soit appuyez sur F8, soit cliquez sur l'icône correspondante dans la barre d'outils.
- Pour fermer une branche parallèle, soit sélectionnez la commande de menu **Insertion ▸ Eléments CONT ▸ Fermer branche,** soit appuyez sur F9, soit cliquez sur l'icône correspondante dans la barre d'outils.
- Une branche parallèle s'ouvre toujours **avant** l'élément CONT sélectionné.
- Une branche parallèle se ferme toujours **après** l'élément CONT sélectionné.
- Pour supprimer une branche parallèle, vous devez commencer par supprimer tous les éléments qu'elle contient. La suppression du dernier élément CONT entraîne la suppression du reste de la branche.

Création de nouvelles branches

Pour ouvrir une nouvelle branche, sélectionnez le point de départ de la branche sous laquelle doit s'insérer la nouvelle, puis appuyez sur la touche F8 (voir figure 3-13).

Création de branches fermées

Pour créer une branche fermée, marquez l'élément CONT avant lequel vous souhaitez ouvrir une branche parallèle. Appuyez sur F8 pour ouvrir la branche parallèle, insérez des éléments CONT et fermez la branche en appuyant sur F9.

Lors de la fermeture de branches parallèles, les éléments vides nécessaires sont complétés. Si besoin est, les branches sont ordonnées de sorte à éviter d'éventuels croisements. Si vous vous placez directement dans la branche parallèle pour réaliser la fermeture d'une branche, la fermeture se fait après le prochain élément CONT possible.

La figure 3-13 illustre un exemple de création de branche parallèle uniquement par sélection de touches de fonction ou d'icônes dans la barre d'outils.

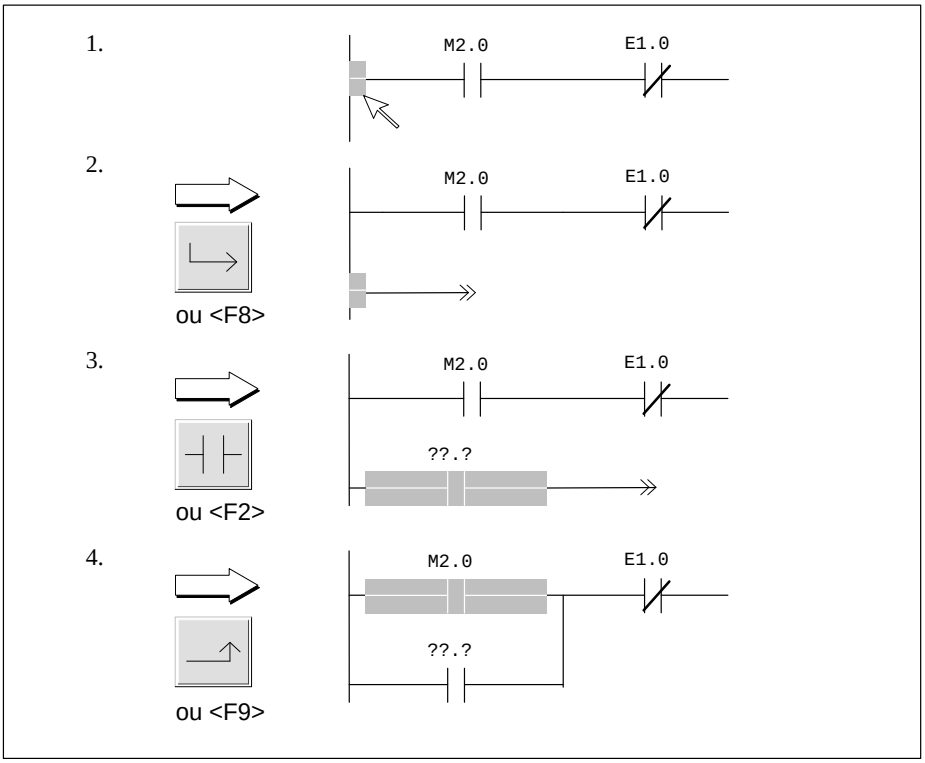


Figure 3-13 Création de branches parallèles dans un réseau CONT

Décomposition de branches parallèles fermées

Vous pouvez décomposer une branche parallèle fermée en créant une séparation au point de jonction entre la branche parallèle et la branche principale.

3.11 Edition d'adresses et de paramètres

Application

Lors de l'insertion d'un élément CONT dans l'éditeur de bloc, la chaîne de caractères ??.? sert de marque de réservation pour adresses et paramètres. Pour qu'une section d'instructions soit exécutable, vous devez définir toutes les adresses et tous les paramètres nécessaires, à l'exception des FB et SFB, des pavés pour temporisation ou pour compteur pour lesquels un paramétrage partiel suffit. L'adressage ou le paramétrage peut être soit absolu, soit symbolique.

Marche à suivre

Pour éditer une adresse ou un paramètre, ouvrez la zone de texte correspondante en cliquant avec le bouton de la souris sur la chaîne de caractères ??.?. Une fois vos données saisies, le programme vérifie la syntaxe. En cas d'erreur, l'adresse ou le paramètre erroné apparaît en rouge. Le programme affiche un message d'erreur dans la barre d'état. Si la syntaxe est correcte, le programme ouvre la prochaine zone de texte à traiter.

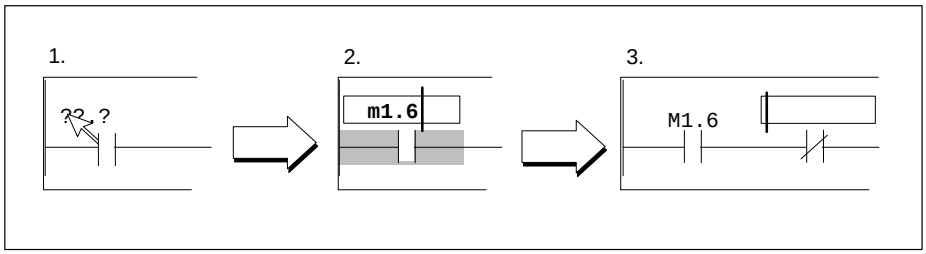


Figure 3-14 Saisie d'adresses pour éléments CONT

Dès que vous vous êtes familiarisé avec les outils d'édition dans le langage CONT, nous vous recommandons de commencer par saisir tous les éléments dans le réseau avant d'affecter une adresse ou un paramètre à chaque élément.

Recherche et correction des erreurs

Pour les reconnaître facilement, les erreurs de syntaxe sont affichées en caractères rouges. Pour atteindre les erreurs aisément, même celles qui se trouvent en dehors de la partie affichée, l'éditeur dispose des deux fonctions suivantes : **Edition ▸ Aller à ▸ Erreur précédente/Erreur suivante** ou d'icônes correspondantes dans la barre d'outils.

La recherche des erreurs n'est pas confinée au réseau actif. Les erreurs sont recherchées dans la section d'instructions entière, c'est-à-dire dans les autres réseaux ainsi que dans la partie non affichée à l'écran. Pour consulter les messages concernant les erreurs, activez la ligne d'état en sélectionnant **Affichage ▸ Barre d'état**.

Pour la correction d'erreurs et pour les modifications, vous pouvez également utiliser le mode de substitution (voir paragraphe 3.13).

3.12 Adressage symbolique

Utilisation de l'adressage symbolique

En langage de programmation CONT, vous pouvez entrer les adresses, les paramètres et les noms de blocs de façon absolue ou symbolique. En sélectionnant **Affichage ▸ Représentation symbolique**, vous pouvez basculer entre la représentation symbolique et absolue.

Définition des mnémoniques

Lorsque vous souhaitez utiliser un mnémonique (nom symbolique) global, vous devez le définir dans la table des mnémoniques.

- Appelez la table des mnémoniques en sélectionnant la commande **Outils ▸ Table des mnémoniques**.
- Appelez une boîte de dialogue en sélectionnant la commande **Outils ▸ Editer les mnémoniques**. Dans cette boîte, vous pouvez définir un nouveau mnémonique ou modifier un mnémonique déjà existant.

Pour de plus amples informations concernant l'édition de mnémoniques, consultez le manuel utilisateur **/231/**.

Représentation

Dans la plupart des cas, il n'est pas nécessaire d'indiquer s'il s'agit d'un mnémonique local ou global. Par contre, si des confusions sont possibles parce que vous avez utilisé des mnémoniques identiques dans la table des mnémoniques et dans la déclaration des variables, par exemple, vous pouvez distinguer les mnémoniques de la façon suivante :

- Les noms symboliques de la table des mnémoniques sont représentés entre guillemets ″ .. ″.
- Les noms de la table de déclaration des variables du bloc sont précédés du signe # pour indiquer qu'il s'agit de variables locales.

Il n'est pas nécessaire d'entrer vous-même les guillemets et le signe #. Si la table de déclaration des variables ou la table des mnémoniques contiennent l'adresse symbolique, ces caractères sont complétés automatiquement après la vérification de syntaxe.

Information mnémonique conviviale

Pour simplifier la programmation avec adressage symbolique, vous pouvez consulter l'adresse absolue correspondant à chaque mnémonique, ainsi que le commentaire de mnémonique. A cet effet, sélectionnez la commande **Affichage ▸ Information mnémoniques**. Une zone d'information est alors affichée après chaque réseau. L'affichage ne peut pas être modifié ; les modifications doivent être apportées dans la table des mnémoniques ou dans la table de déclaration des variables.

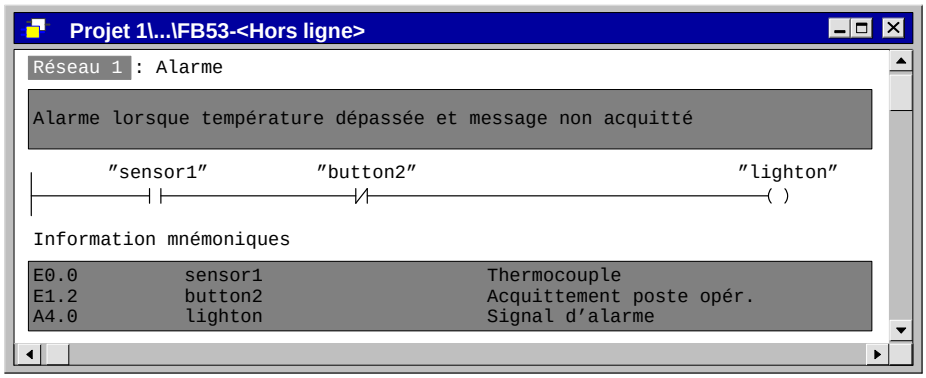


Figure 3-15 Information sur les mnémoniques en CONT

Lors de l'impression du bloc, la représentation actuelle à l'écran est éditée avec le commentaire d'instruction ou le commentaire de mnémonique.

Nota

La table des mnémoniques n'est pas chargée dans la CPU en même temps que le programme correspondant. En d'autres termes, si vous souhaitez traiter un programme utilisateur pour lequel vous ne disposez pas du programme d'origine dans le système de développement (PG ou PC), vous ne disposez pas non plus des mnémoniques d'origine.

3.13 Modification en mode de substitution

Substitution d'adresses et de paramètres

L'éditeur de bloc dispose d'un mode de substitution vous permettant de modifier facilement des adresses et paramètres. Pour passer d'un mode à l'autre, appuyez sur la touche d'insertion INSER. Dans les zones de texte, vous pouvez modifier vos entrées d'adresse ou de paramètre par substitution.

Substitution d'éléments CONT

Le mode de substitution permet de remplacer aisément un élément CONT par un autre élément de type identique. Ce faisant, toutes les liaisons booléennes et tous les paramétrages sont conservés.

L'avantage de cette méthode est que vous n'avez pas à saisir à nouveau les adresses et les paramètres. En respectant la règle d'identité de type des éléments, vous pouvez, par exemple, remplacer contact à fermeture par contact à ouverture, bascule « mise à 1, mise à 0 » par bascule « mise à 0, mise à 1 » ou différents temporisateurs entre eux.

Pour remplacer des éléments CONT existants, sélectionnez-les et passez en mode de substitution à l'aide de la touche INSER. L'élément est modifié par substitution dès que vous procédez à la saisie d'un élément de type identique.

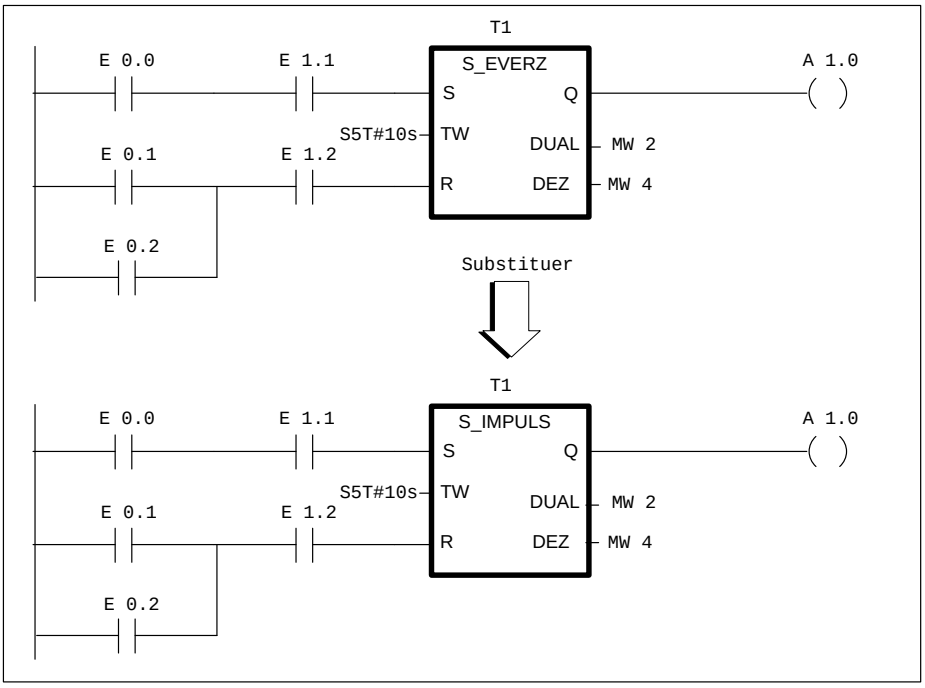


Figure 3-16 Substitution de pavés

Cas exceptionnel : décomposition de jonction

Lorsque dans une branche, une branche parallèle se ferme et qu'une autre s'ouvre simultanément, une jonction se crée. Vous pouvez décomposer cette jonction en mode de substitution en sélectionnant le point de liaison supérieur ou inférieur et en insérant l'élément CONT.

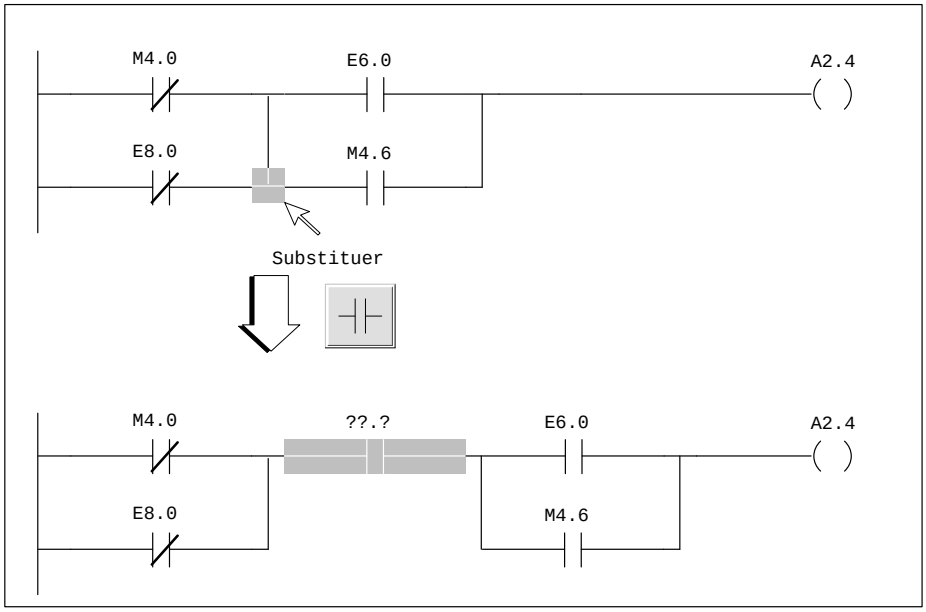


Figure 3-17 Décomposition d'une jonction

Nota

Vous pouvez corriger commentaires et titres en mode d'insertion.

3.14 Saisie de titres et de commentaires

Aperçu

Dans la section d'instructions d'un bloc de code, vous pouvez saisir le titre du bloc, les titres de réseau ainsi que les commentaires de bloc. Les entrées sont facultatives et n'ont pas d'incidence sur le déroulement du programme.

Saisie de titres de bloc et de réseau

Pour saisir le titre du bloc ou du réseau, positionnez le curseur sur les trois points d'interrogation à droite du nom de bloc ou de réseau (par exemple Réseau 1 : ???). Un champ d'édition est affiché, dans lequel vous pouvez entrer un titre de 64 caractères au maximum.

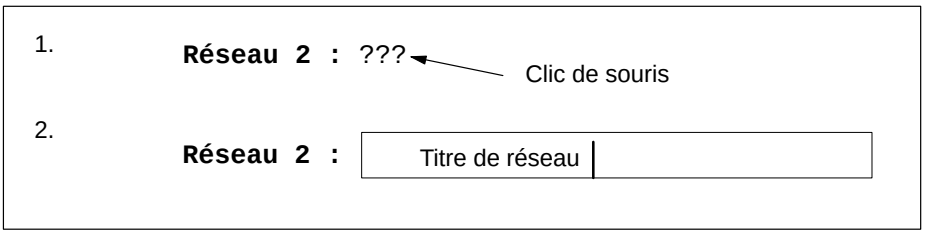


Figure 3-18 Saisie de titres

Saisie de commentaires

Vous pouvez afficher ou occulter la zone de commentaire grise en sélectionnant **Affichage ▸ Commentaire**. En effectuant un double-clic sur une telle zone, vous affichez la zone de texte dans laquelle vous pouvez alors entrer vos commentaires. Vous disposez de 64 Ko par bloc pour les commentaires de bloc et de réseau.

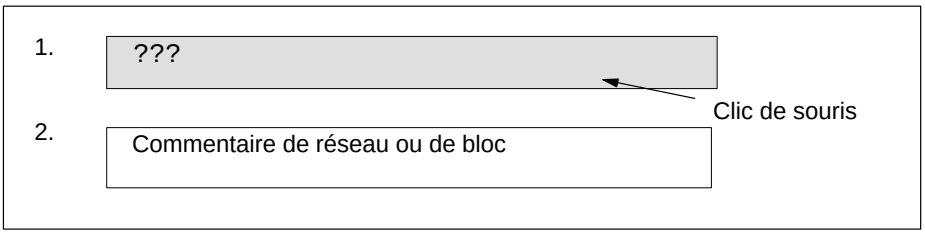


Figure 3-19 Saisie de commentaires

Nota

Les commentaires ne sont pas chargés dans la CPU en même temps que le bloc correspondant. En d'autres termes, si vous chargez depuis une CPU un bloc pour lequel vous ne disposez pas du bloc d'origine dans le système de développement (PG ou PC), vous ne pourrez ni consulter ni modifier les commentaires d'origine.

Création de blocs de données et de types de données utilisateur

4

Présentation

Les blocs de données sont un élément important de votre programme utilisateur, étant donné qu'ils contiennent les données du programme utilisateur. Ce chapitre décrit la création de ces blocs de données.

Les types de données utilisateur (UDT) ne sont pas indispensables à la programmation. Cependant, si vous créez souvent des programmes pour des tâches analogues, vous utiliserez bientôt vos « propres » types de données, car ils représentent un gain de temps considérable.

Contenu de ce chapitre

Paragraphe	Thème	Page
4.1	Création de blocs de données, aperçu	4-2
4.2	Choix de la méthode de création	4-4
4.3	Edition de la table de déclaration	4-5
4.4	Edition des valeurs de données en cours	4-6
4.5	Création de types de données utilisateur	4-8

4.1 Création de blocs de données, aperçu

Blocs de données

Les blocs de données (DB) vous permettent de gérer les données, ce qui explique que vous ne disposiez pas de section d'instructions. Lors de la programmation, vous devez donc tenir compte des sections suivantes :

- **Table de déclaration** : cette table vous permet de déterminer la structure des données du bloc de données.
- **Propriétés des blocs de données** : ces propriétés comprennent des informations complémentaires, telles que horodatage, langage de création ou indication du chemin, inscrites par le système. Il est également possible d'entrer des données concernant le nom, la famille, la version et l'auteur (voir chapitre 5) ainsi que d'affecter des attributs système aux blocs.

Catégories de DB

Dans un programme utilisateur, on distingue les catégories de blocs de données suivantes :

- Les **blocs de données globaux** auxquels tous les blocs du programme peuvent accéder et qui conservent les données après la fermeture du bloc.

  Si vous avez besoin de plusieurs blocs de données globaux de même structure, vous pouvez facilement créer ces derniers en utilisant un type de données utilisateur (UDT). Il s'agit alors de **blocs de données associés à un UDT**.
- Les **blocs de données d'instance** sont affectés à des blocs fonctionnels et structurés en fonction des variables déclarées dans le FB. La création d'un DB d'instance nécessite donc un FB existant. Il s'agit de **blocs de données associés à un bloc fonctionnel**.

Méthodes de création des blocs de données

La méthode utilisée pour la création d'un DB varie en fonction de la catégorie du DB que vous souhaitez créer.

Vous pouvez créer un **bloc de données global** suivant les méthodes suivantes :

- Définir la structure pour un DB isolé.
  Vous devez définir les variables et les types de données pour le DB dans l'ordre souhaité. La structure s'applique alors uniquement à ce bloc de données.
- Définir la structure du DB à l'aide d'un type de données utilisateur.
  Dans ce cas, la structure de données de l'UDT conditionne la structure du DB. Un UDT peut être affecté à plusieurs DB.

Pour créer un **DB d'instance** :

- Associez le DB à un FB existant. Dans ce cas, la section de déclaration du FB détermine la structure du DB. Plusieurs DB d'instance peuvent être affectés à un même FB.

Nota

Si vous modifiez ultérieurement la déclaration de variables d'un FB, vous devez générer à nouveau les DB d'instance correspondants si vous voulez que ces blocs restent compatibles. Cela vaut également pour les DB générés par le biais d'une affectation à un UDT.

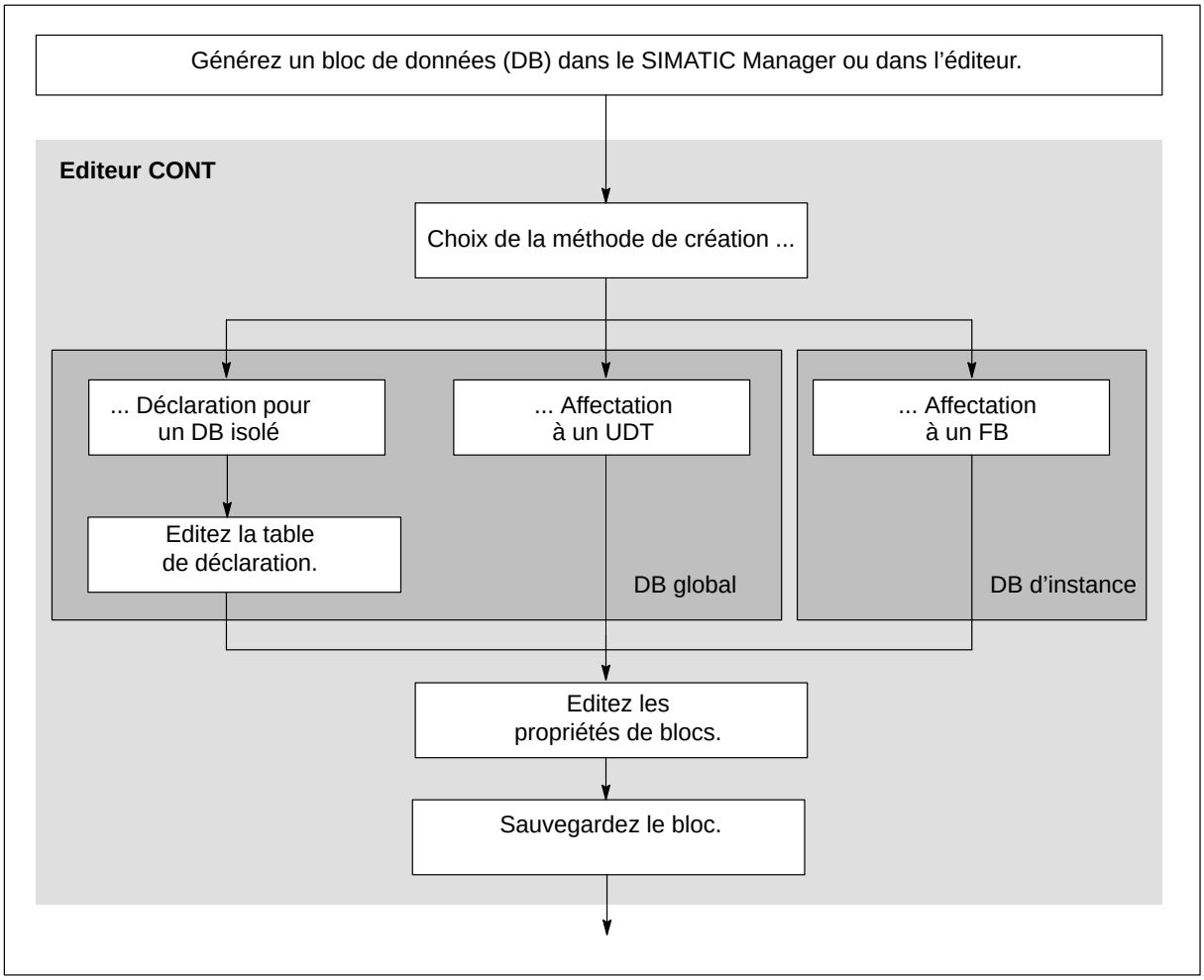

Générez un bloc de données (DB) dans le SIMATIC Manager ou dans l'éditeur.
Editeur CONT
Choix de la méthode de création ...
... Déclaration pour un DB isolé
... Affectation à un UDT
... Affectation à un FB
Editez la table de déclaration.
DB global
DB d'instance
Editez les propriétés de blocs.
Sauvegardez le bloc.

Figure 4-1 Marche à suivre lors de la programmation de blocs de données

4.2 Choix de la méthode de création

Marche à suivre

Lors de la génération d'un bloc de données dans le SIMATIC Manager ou dans l'éditeur CONT, vous devez indiquer la méthode de création. La boîte de dialogue suivante apparaît alors à l'écran :

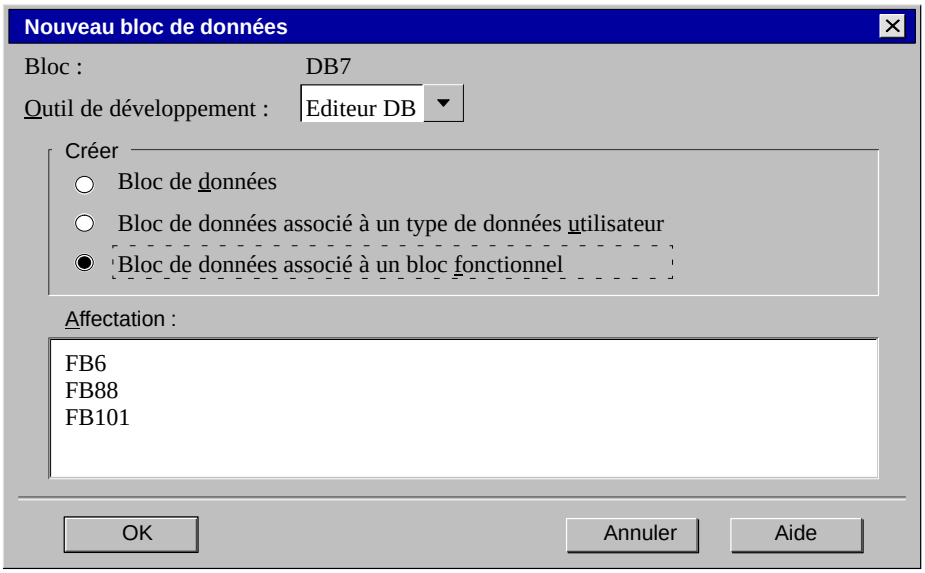


Figure 4-2 Choix de la méthode de création et affectation à un FB ou à un UDT

Si vous voulez générer le bloc de données avec un UDT ou comme bloc de données d'instance d'un bloc fonctionnel, vous devez sélectionner celui-ci dans la liste dans laquelle figurent tous les UDT ou blocs fonctionnels du programme. L'existence de l'UDT ou du bloc fonctionnel correspondant constitue donc une condition sine qua non.

Suite de la procédure

La suite de la procédure dépend du mode de création du bloc de données, par affectation ou par déclaration individuelle.

- L'UDT ou le bloc fonctionnel déterminant clairement la structure du bloc de données, la création du bloc est donc terminée. Pour contrôler vos données, utilisez la table de déclaration affichée que vous ne pouvez cependant pas traiter.
- Par contre, lors de la création individuelle d'un bloc de données global, vous devez éditer la table de déclaration dans laquelle vous indiquez les noms de variables ainsi que, si vous le désirez, la valeur initiale et les commentaires éventuels (voir paragraphe 4.3).

4.3 Edition de la table de déclaration

Objectif de la vue des déclarations

Lors de la création individuelle de blocs de données globaux ou lors de la création d'UDT, vous devez entrer des indications relatives aux éléments (variables) du bloc de données ou de l'UDT et à leur type de données. Utilisez, à cet effet, la vue des déclarations de la table de déclaration à laquelle vous accédez, le cas échéant, via la commande **Affichage ▸ Vue des déclarations**.

Cette opération devient superflue s'il s'agit de blocs de données générés par affectation d'UDT ou de bloc fonctionnel, la déclaration étant alors reprise de l'UDT ou du bloc fonctionnel correspondant.

Structure de la table dans la vue des déclarations

Dans la vue des déclarations d'un bloc de données figurent l'adresse, le type de déclaration (seulement pour les DB d'instance), le nom de la variable (mnémonique), le type de données, la valeur initiale et le commentaire. Voir l'exemple ci-dessous à la figure 4-3 :

DB15 - <Hors ligne>

Adresse	Nom	Type	Valeur initiale	Commentaire
0.0		STRUCT		
+0.0	speed	INT	100	Régime maximum
+2.0	runtime	DINT	L#0	
+6.0	history	REAL	0.000000e+000	
+10.0	motor_on	BOOL	FALSE	
+10.1	motor_off	BOOL	FALSE	
=12.0		END_STRUCT		

Figure 4-3 Déclaration d'un bloc de données

Les colonnes ont la même signification que dans la table de déclaration des blocs de code (voir paragraphe 3.3).

Marche à suivre

Lors de la saisie d'une nouvelle déclaration, vous entrez successivement, après le type de déclaration souhaité, le nom de la variable, le type de données, la valeur initiale (facultative) et le commentaire (facultatif). Pour déplacer le curseur, utilisez la touche TAB ou ENTREE. Une fois la ligne renseignée, une adresse est affectée à la variable.

Après chaque traitement d'une zone de la table, le système effectue une **vérification de la syntaxe** qui, le cas échéant, fait apparaître les erreurs en rouge. Vous n'êtes pas obligé de les rectifier immédiatement ; vous pouvez en effet poursuivre le traitement et corriger les erreurs ultérieurement.

Nota

L'édition de la vue des déclarations correspond à l'édition de la table de déclaration des variables de blocs de code (voir paragraphe 3.4). Les fonctions d'édition et la marche à suivre sont identiques, même lors de la saisie de tableaux ou de structures.

4.4 Edition des valeurs de données en cours

Valeur initiale – valeur en cours

Lors de la création d'un bloc de données, la valeur initiale – facultative – est reprise en tant que valeur en cours de la variable quand le bloc de données est enregistré pour la première fois. Lorsque le programme utilisateur accède pour la première fois à ces données, il utilise cette valeur, sauf si vous avez déterminé explicitement une autre valeur en cours pour la variable dans le programme utilisateur.

Les valeurs en cours des variables sont modifiées par les blocs de code qui y accèdent en écriture lors de l'exécution du programme utilisateur. Vous avez la possibilité de consulter et de modifier les valeurs en cours des variables.

Vue des données des blocs de données

Vous consultez et modifiez les données des blocs de données dans la vue des données. Ouvrez un bloc de données et passez, le cas échéant, dans la vue des données en utilisant la commande **Affichage ▸ Vue des données**.

Seule la colonne complémentaire « Valeur en cours » distingue la vue des données de la vue des déclarations d'un bloc de données. S'il s'agit de variables dotées d'un type de données complexe, la vue des données liste les éléments individuellement, avec leur mnémonique entier, afin que vous puissiez afficher et éditer la valeur en cours de chaque élément (voir figure 4-4).

DB17 - <Hors ligne>

Adresse	Nom	Type	Valeur initiale	Valeur en cours	Commentaire
0.0	motor.speed	INT	100	89	Régime maximum
2.0	motor.runtime	DINT	L#0	L#0	
6.0	motor.history	REAL	0.000000e+000	0.000000e+000	
10.0	motor.motor_on	BOOL	FALSE	TRUE	
10.1	motor.motor_off	BOOL	FALSE	FALSE	
12.0	field[1]	INT	0	7	
14.0	field[2]	INT	0	4	
16.0	field[3]	INT	0	8	

Figure 4-4 Vue des données d'un DB

Valeur en cours affichée

Il s'agit de la valeur de la variable en vigueur au moment de l'ouverture du DB ou de la dernière valeur modifiée et sauvegardée.

Nota

La valeur en cours n'est pas mise à jour cycliquement dans le cas de blocs de données ouverts en ligne.

Modification et réinitialisation de valeurs en cours

Les valeurs en cours peuvent être modifiées par substitution dans la colonne correspondante. Les valeurs indiquées doivent être compatibles avec le type de données.

La commande **Edition ▸ Initialisation de bloc de données** permet de réinitialiser le bloc de données entier. Cette fonction remplace les valeurs en cours des variables par les valeurs initiales, déterminées dans la vue des déclarations ou par le biais du FB ou de l'UDT associé.

Sauvegarde des valeurs en cours

Pour valider les valeurs en cours, vous devez les sauvegarder :

- Pour sauvegarder hors ligne les valeurs en cours que vous avez modifiées, sélectionnez la commande **Fichier ▸ Enregistrer** ou cliquez sur l'icône « Enregistrer ». Même si vous avez ouvert le bloc de données en ligne, seul est sauvegardé à nouveau le bloc de données existant hors ligne.
- Pour charger les valeurs modifiées dans la CPU, sélectionnez la commande **Système cible ▸ Charger** ou cliquez sur l'icône correspondante dans la barre d'outils.

4.5 Création de types de données utilisateur

Application

Les types de données utilisateur sont des structures de données (STRUCT) créées par l'utilisateur et sauvegardées comme bloc. Vous pouvez les utiliser dans le programme utilisateur entier avec leur nom de bloc absolu ou symbolique. Ils peuvent :

- être utilisés comme des **types de données** élémentaires ou complexes dans la table de déclaration des variables des blocs de code (FC, FB, OB) ou dans les blocs de données (DB) ;
- servir de **modèle pour la création de blocs de données** ayant la même structure.

Marche à suivre

La figure 4-5 montre la marche à suivre pour la création d'un bloc de données utilisateur (UDT).

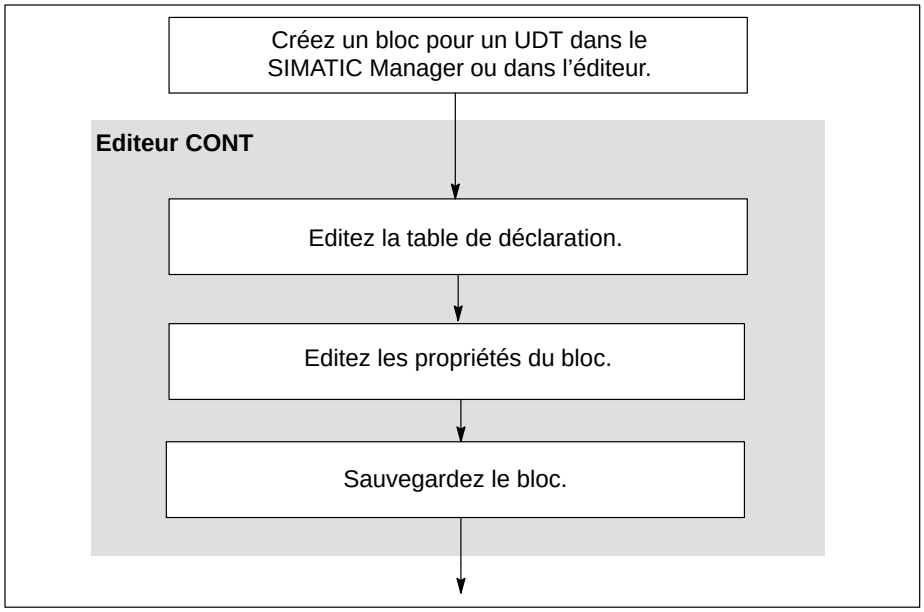
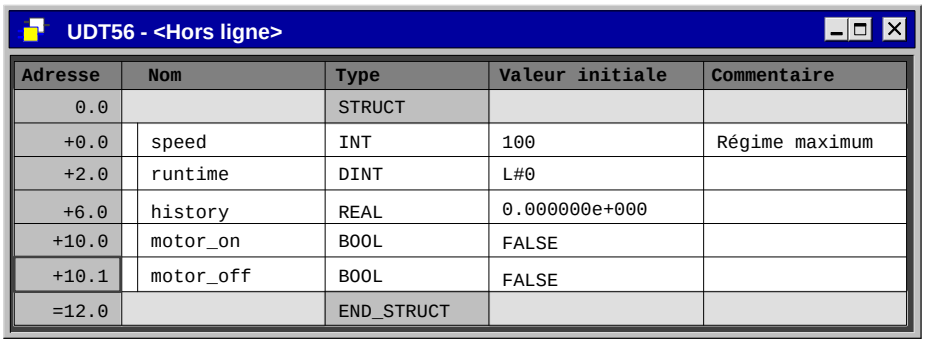

Créez un bloc pour un UDT dans le SIMATIC Manager ou dans l'éditeur.
Editeur CONT
Editez la table de déclaration.
Editez les propriétés du bloc.
Sauvegardez le bloc.

Figure 4-5 Création de types de données utilisateur (UDT)

Edition de la table de déclaration

Après avoir créé ou ouvert un UDT dans le SIMATIC Manager ou dans l'éditeur incrémental, vous voyez apparaître une table de déclaration dans laquelle vous devez indiquer la structure du type de données.



UDT56 - <Hors ligne>

Adresse	Nom	Type	Valeur initiale	Commentaire
0.0		STRUCT		
+0.0	speed	INT	100	Régime maximum
+2.0	runtime	DINT	L#0	
+6.0	history	REAL	0.000000e+000	
+10.0	motor_on	BOOL	FALSE	
+10.1	motor_off	BOOL	FALSE	
=12.0		END_STRUCT		

Figure 4-6 Déclaration d'un type de données utilisateur (UDT)

La première et la dernière lignes de la vue des déclarations pour un UDT contiennent respectivement STRUCT et END_STRUCT pour indiquer le début et la fin de l'UDT ; ces lignes sont protégées.

La table contient deux lignes vierges pour vous permettre de saisir les deux premières variables. Au besoin, entrez les valeurs initiales et ajoutez les commentaires. Vous pouvez insérer des lignes vierges en appuyant sur la touche d'entrée à la fin d'une ligne ou avec la commande **Insertion ▸ Ligne de déclaration ▸ Avant la sélection /Après la sélection**.

Nota

Ce traitement correspond à l'édition de structures dans la section de déclaration de blocs de code ou de données.

Edition des propriétés de bloc et test du programme

5

Présentation

Après avoir créé et programmé les blocs de données et des blocs de code, vérifiez et éditez les propriétés de bloc. Elles vous permettent, par la suite, d'identifier exactement un bloc et de déterminer son origine, ce qui peut s'avérer très utile pour remédier aux erreurs éventuelles.

Dans l'éditeur CONT, il est possible de tester chaque bloc traité dans un programme utilisateur au niveau de la CPU. Au cours des tests, vous pouvez suivre à l'écran le flux d'énergie dans les réseaux. Cette possibilité de test, encore appelée visualisation d'état de programme, vous aide, pendant la mise en service, à contrôler les déroulements de programme importants ainsi qu'à remédier aux erreurs.

Contenu de ce chapitre

Paragraphe	Thème	Page
5.1	Edition des propriétés de bloc	5-2
5.2	Test du programme CONT, aperçu	5-5
5.3	Paramétrage de l'affichage pour visualisation d'état de programme	5-6
5.4	Paramétrage des conditions de déclenchement	5-7
5.5	Sélection d'environnement de test et lancement ou arrêt de la visualisation d'état de programme	5-8

5.1 Edition des propriétés de bloc

Aperçu

Les propriétés de bloc contiennent des informations supplémentaires relatives à un bloc. Par ailleurs, vous pouvez entrer vous-même le nom, la famille, la version et l'auteur du bloc et aussi affecter des attributs système au bloc. D'autres informations et données statistiques entrées par le système, et que vous ne pouvez pas éditer, sont également affichées (cf. figure 5-1).

Les propriétés de bloc livrent des informations importantes, telles que le type de bloc, l'espace mémoire requis ainsi que la date et l'heure de la dernière modification. Ce type d'information vous aide à rechercher l'origine des erreurs lorsque la capacité mémoire est insuffisante ou lors de conflits d'horodatage, par exemple.

Marche à suivre

L'affichage et l'édition des propriétés de bloc peut s'effectuer en mode interactif. Pour ce faire, procédez comme suit :

- Dans le SIMATIC Manager, sélectionnez le bloc et exécutez la commande **Edition ▸ Propriétés de l'objet**.
- Lorsque le bloc est ouvert dans l'éditeur CONT, sélectionnez la commande **Fichier ▸ Propriétés**.

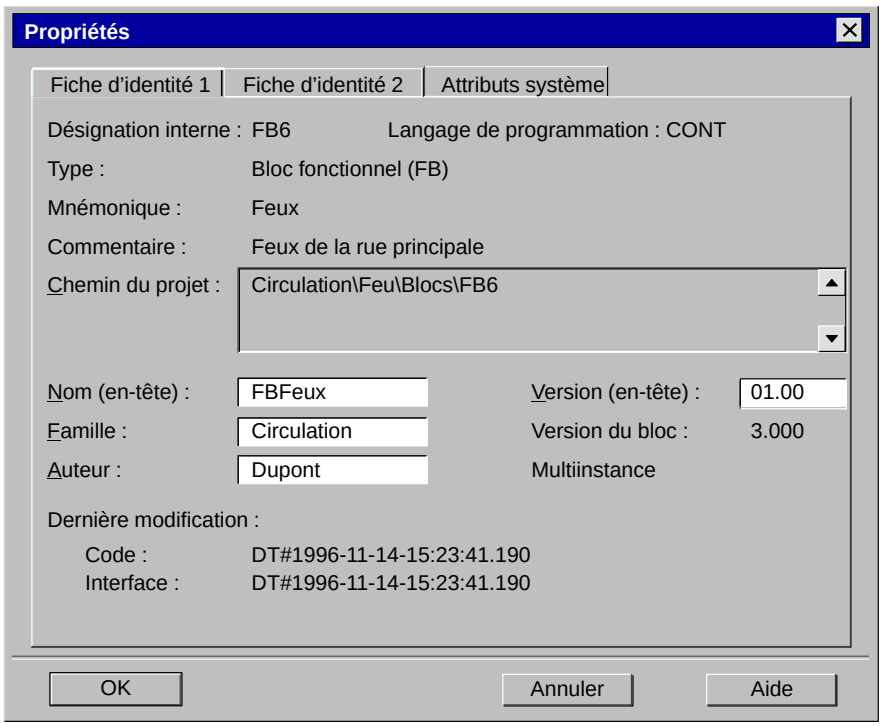


Figure 5-1 Boîte de dialogue des propriétés de bloc

Vous pouvez entrer les indications ci-après dans les pages d'onglet « Fiche d'identité 1 », « Fiche d'identité 2 » et « Attributs système ».

Nom et famille du bloc

Le nom et la famille du bloc vous permettent de classifier les blocs que vous avez créés. Ainsi, vous pouvez affecter tous les blocs qui servent à programmer une régulation à une famille portant le même nom, par exemple.

C'est lors de l'appel ultérieur du bloc dans la section d'instructions d'un autre bloc en CONT que ces informations sont utiles. Dans la boîte de dialogue « Insérer un élément CONT », le système affiche automatiquement le nom et la famille du bloc sélectionné de sorte que l'utilisation du bloc est plus facilement identifiable.

Version du bloc

En fonction des attributs de bloc, vous pouvez identifier la version de STEP 7 qui a servi à la création du bloc. Vous devez convertir les blocs de la version 1 pour pouvoir les intégrer à un programme de version 2. A cet effet, sélectionnez la commande de menu **Fichier ▸ Ouvrir un ancien projet de version 1...** dans le SIMATIC Manager.

Vous ne pouvez pas utiliser de blocs de la version 1 avec des multi-instances. Si c'est nécessaire, vous devrez d'abord générer un fichier source à partir de ces blocs, puis convertir ce fichier en blocs de la version 2.

Pour de plus amples informations à ce sujet, veuillez consulter le guide de l'utilisateur **/231/**.

Attributs de bloc

L'onglet « Fiche d'identité 2 » affiche les attributs du bloc.

- L'attribut « **DB protégé en écriture dans l'AS** » signale que le bloc est protégé en écriture. Il est utile d'activer cet attribut dans le cas de blocs de données dont les valeurs restent constantes et ne doivent pas être modifiées. Le DB concerné doit exister sous la forme d'une source LIST.
- L'attribut « **Protection KNOW HOW** » indique que l'accès au bloc est protégé. Par conséquent :
  - Vous ne pouvez pas consulter la section d'instructions du bloc.
  - Dans la table de déclaration des variables n'apparaissent ni variables temporaires, ni variables statiques.
  - Vous ne pouvez pas générer de fichier source LIST à partir du bloc.
  - Vous ne pouvez pas éditer les propriétés du bloc.
- L'attribut « **Bloc standard** », lorsqu'il est activé, indique qu'il s'agit d'un bloc standard Siemens protégé. Le système le signale en affichant un message correspondant en bas de l'écran, à gauche.
- L'attribut « **Unlinked** » ne peut être utilisé que pour des blocs de données. Il indique que le bloc de données n'est pas chargé de la mémoire de chargement dans la mémoire de travail de la CPU. L'accès aux blocs de données dans la mémoire de chargement s'effectue par SFC copiant uniquement le contenu des DB dans la mémoire de travail. Ainsi, vous optimisez l'utilisation de la mémoire de travail puisqu'elle contient uniquement les données utiles au moment de l'exécution.

Nota

Vous ne pouvez indiquer pour votre bloc les attributs de protection de bloc, de protection en écriture et de non-intégration (unlinked) que lorsque vous le programmez comme fichier source LIST. Si vous avez généré un bloc en CONT, passez au langage de programmation LIST en sélectionnant la commande **Affichage ▶ LIST**. Convertissez le bloc en un fichier source dans lequel vous pourrez saisir les attributs. Après reconversion du fichier source en un bloc, ce dernier se présente sous la forme protégée correspondante.

Pour plus d'informations à ce sujet, veuillez consulter le manuel pour LIST **/232/**.

Attributs système pour blocs

Dans la page d'onglet « Attributs système », vous pouvez affecter à des blocs les attributs système suivants pour la configuration du système de conduite et le diagnostic du processus.

Tableau 5-1 Attributs système pour la configuration du système de conduite

Attribut	Valeur	Vous affectez cet attribut ...	Type de bloc autorisé
S7_m_c	true, false	lorsque le bloc doit être contrôlé ou commandé par un appareil de contrôle-commande.	FB, SFB
S7_tasklist	'taskname1', 'taskname2', etc.	lorsque le bloc doit être appelé dans des blocs autres que les blocs d'organisation cycliques (par exemple, dans des blocs d'erreur ou de démarrage).	FB, SFB, FC, SFC
S7_block-view	big, small	lorsque le bloc doit être représenté en grand ou petit format.	FB, SFB, FC, SFC

Tableau 5-2 Attributs système pour le diagnostic du processus

Attribut	Valeur	Vous affectez cet attribut ...	Type de bloc autorisé
S7_pdiag	true, false	lorsque le bloc doit générer des informations significatives pour le diagnostic du processus.	FB, FC, OB, DB
S7_pdiag_ unit	true, false	lorsque le bloc doit générer des informations significatives pour le diagnostic du processus et qu'une unité doit être surveillée.	UDT
S7_pdiag_ motion	true, false	lorsque le bloc doit générer des informations significatives pour le diagnostic du processus et qu'un mouvement doit être surveillé.	UDT

5.2 Test du programme CONT, aperçu

Possibilité de test

Pour tester votre programme CONT, il vous suffit de suivre le flux d'énergie dans les réseaux d'un bloc. L'affichage de l'état du programme est actualisé de façon cyclique.

Conditions préliminaires

Pour pouvoir afficher l'état de programme, les conditions préliminaires suivantes doivent être remplies :

- sauvegarde correcte du bloc et chargement ultérieur dans la CPU,
- CPU en fonctionnement et exécution du programme utilisateur,
- ouverture du bloc en mode en ligne.

Marche à suivre

La figure 5-2 illustre la marche à suivre pour visualiser l'état du programme :

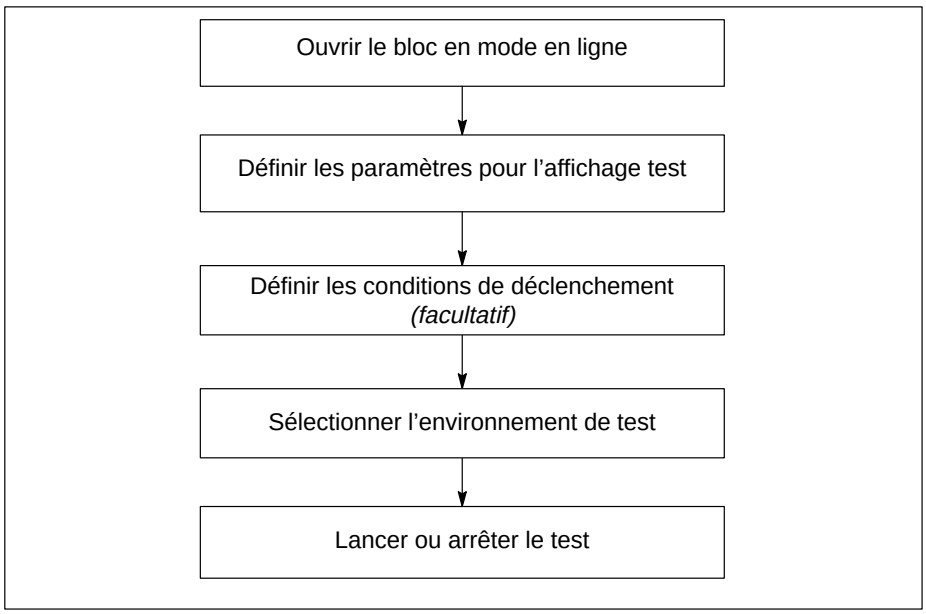

Ouvrir le bloc en mode en ligne
Définir les paramètres pour l'affichage test
Définir les conditions de déclenchement
(facultatif)
Sélectionner l'environnement de test
Lancer ou arrêter le test

Figure 5-2 Marche à suivre pour tester des blocs de code en CONT

5.3 Paramétrage de l’affichage pour visualisation d’état de programme

Marche à suivre

Avant de lancer la visualisation d’état de programme pour un bloc en CONT, définissez le mode d’affichage du flux d’énergie. En sélectionnant la commande **Outils** ▸ **Paramètres**, vous pouvez ouvrir la page d’onglet « CONT » nécessaire.

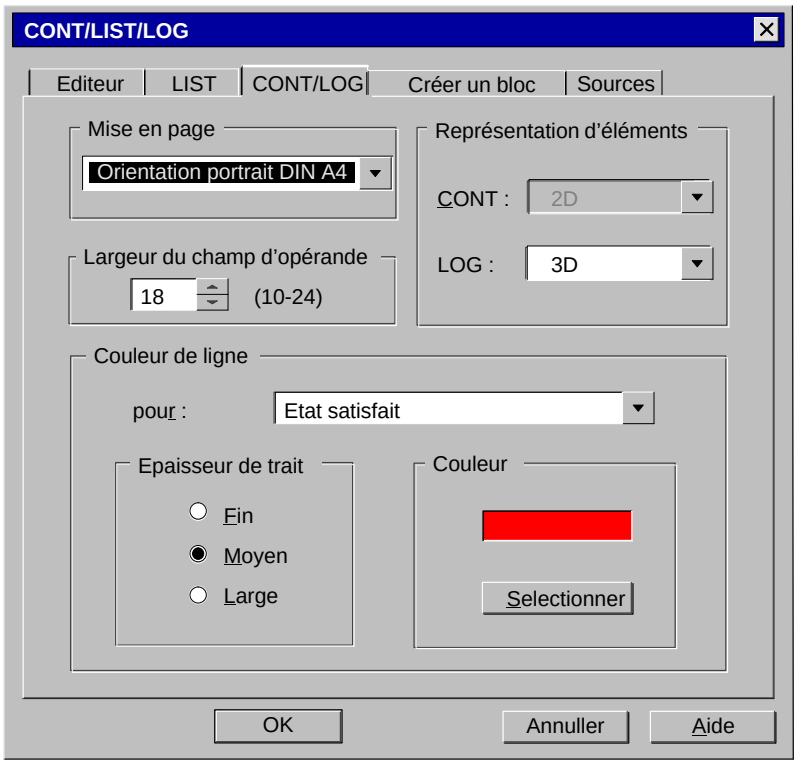


Figure 5-3 Paramétrage de l’affichage pour la visualisation d’état de programme en CONT

Sélectionnez, dans cette page, la couleur et l’épaisseur du trait pour les deux cas suivants :

- « Etat non satisfait » : les conditions tout au long du trajet du courant ne sont pas remplies. Le courant ne circule pas (lignes en pointillés).
- « Etat satisfait » : les conditions tout au long du trajet du courant sont remplies. Le courant circule (lignes continues).

5.4 Paramétrage des conditions de déclenchement

Origine

Les conditions de déclenchement vous permettent de paramétrer l'environnement d'appel du bloc à tester. La fonction de test n'est exécutée que si la condition de déclenchement paramétrée est remplie.

Marche à suivre

Vous pouvez paramétrer les conditions de déclenchement en sélectionnant la commande **Test ▸ Conditions d'appel**.

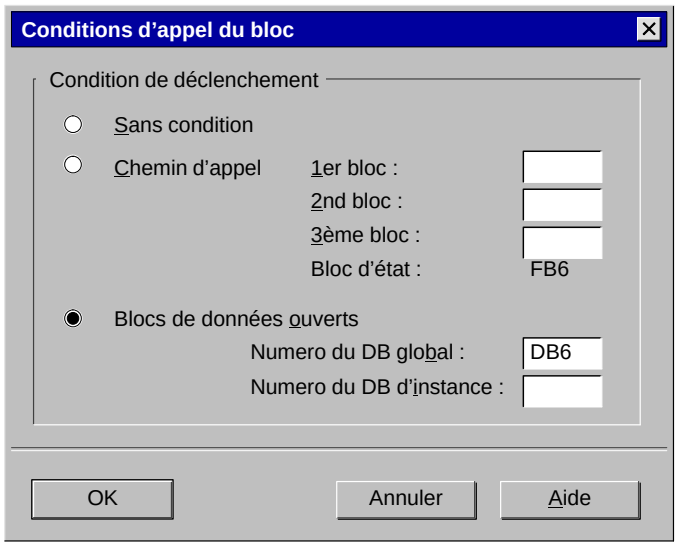


Figure 5-4 Définition de la condition de déclenchement

Signification des conditions de déclenchement

A chacune des trois possibilités de paramétrage correspond une signification différente :

- Pas de condition de déclenchement : l'environnement d'appel est sans importance pour le bloc à tester. Toutefois, si vous appelez le même bloc à différents endroits du programme, vous ne pouvez pas reconnaître pour quel appel l'état s'affiche.
- Chemin d'appel : vous avez la possibilité d'indiquer ici le chemin par lequel le bloc à tester doit être appelé pour que l'état soit enregistré. Vous pouvez entrer les trois derniers niveaux d'appel avant l'accession au bloc de test.
- Blocs de données ouverts : vous pouvez définir ici l'environnement d'appel en entrant un ou deux blocs de données. L'état est enregistré lorsque le bloc à tester est appelé avec le bloc de données indiqué.

5.5 Sélection d'environnement de test et lancement ou arrêt de la visualisation d'état de programme

Sélection d'environnement de test

Il existe deux situations en ligne pour pouvoir tester votre programme :

- En environnement de test « **Processus** », vous testez votre programme en ligne dans les conditions correspondant au processus. Dans ce cas, l'état des instructions figurant dans des boucles programmées s'exécutant plusieurs fois pendant le cycle est déterminé uniquement au premier passage du programme. Ce mode représente une faible charge du temps de cycle.
- En environnement de test **« Laboratoire »**, vous testez également en ligne, mais dans des conditions de laboratoire. Dans ce cas, l'état des instructions est déterminé à chaque passage du programme. Ce mode augmente considérablement la charge du temps de cycle, étant donné que le temps de cycle est fonction du nombre de passages du programme et du nombre d'instructions visualisées.

Pour sélectionner l'environnement de test, utilisez la commande **Test ▸ Environnement de test ▸ Laboratoire/Processus**.

Lancement et arrêt de la visualisation d'état de programme

Pour lancer ou arrêter la visualisation d'état de programme, sélectionnez la commande **Test ▸ Visualiser.** L'état de programme affiché s'applique uniquement à la partie visible dans l'éditeur.

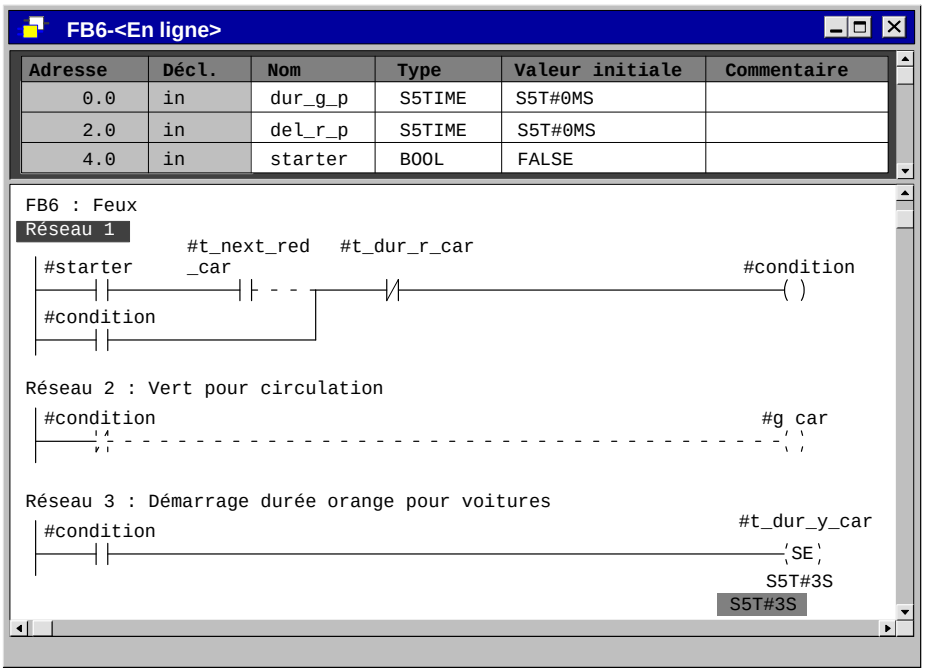


Figure 5-5 Etat de programme en CONT (exemple)

Vérification du temps de cycle

L'activation du mode de test augmente le temps de cycle. En cas de dépassement du temps de cycle paramétré, la CPU passe à l'état de fonctionnement « Arrêt » (STOP), à moins que vous n'ayez programmé un OB 80.

Pour afficher et vérifier le temps de cycle paramétré et le temps de cycle en cours, sélectionnez la commande **Système cible ▸ Etat du module**. Si besoin est, vous pouvez modifier, à des fins de test, le temps de cycle maximum dans les propriétés de la CPU lors du paramétrage du matériel.

Deuxième partie : Description du langage

Structure et éléments de CONT 6

Adressage 7

Opérations combinatoires sur bits 8

Opérations de temporisation 9

Opérations de comptage 10

Opérations arithmétiques sur nombres entiers 11

Opérations arithmétiques sur nombres réels 12

Opérations de comparaison 13

Opérations de transfert et de conversion 14

Opérations combinatoires sur mots 15

Opérations de décalage et de rotation 16

Opérations sur blocs de données 17

Opérations de saut 18

Opérations sur bits d’état 19

Opérations de gestion d’exécution de programme 20

Structure et éléments de CONT

6

Contenu de ce chapitre

Paragraphe	Thème	Page
6.1	Eléments et pavés	6-2
6.2	Logique booléenne et tables de vérité	6-6
6.3	Signification des registres de la CPU dans les instructions	6-12

6.1 Eléments et pavés

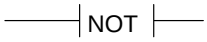
Instructions CONT

Les instructions CONT se composent d'éléments et de pavés reliés graphiquement en réseaux. Ils peuvent être classés en différents groupes.

Opérations sous forme d'éléments

STEP 7 représente certaines opérations CONT sous forme d'éléments individuels ne nécessitant ni opérande, ni paramètres (voir tableau 6-1).

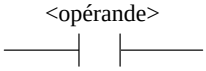
Tableau 6-1 Opération CONT sous forme d'élément sans opérande, ni paramètres

Elément	Nom	Paragraphe dans ce manuel
	Inverser RLG	8.6

Opérations sous forme d'éléments avec opérande

STEP 7 représente certaines opérations CONT sous forme d'éléments individuels nécessitant un opérande (voir tableau 6-2).
Reportez-vous au chapitre 7 pour plus d'informations sur l'adressage.

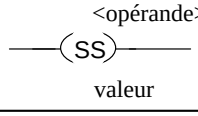
Tableau 6-2 Opération CONT sous forme d'élément avec opérande

Elément	Nom	Paragraphe dans ce manuel
	Contact à fermeture	8.2

Opérations sous forme d'éléments avec opérande et valeur

STEP 7 représente certaines opérations CONT sous forme d'éléments individuels nécessitant un opérande et une valeur (telle que valeur de comptage ou de temps ; voir tableau 6-3).
Reportez-vous au chapitre 7 pour plus d'informations sur l'adressage.

Tableau 6-3 Opération CONT sous forme d'élément avec opérande et valeur

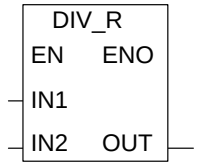
Elément	Nom	Paragraphe dans ce manuel
	Temporisation sous forme de retard à la montée mémorisé	8.16

Opérations sous forme de pavés avec paramètres

STEP 7 représente certaines opérations CONT sous forme de pavés avec des lignes indiquant les entrées et les sorties (voir tableau 6-4). Les entrées se situent du côté gauche du pavé et les sorties du côté droit. Vous indiquez les paramètres d'entrée. Pour les paramètres de sortie, vous précisez les adresses où le logiciel STEP 7 mettra les informations de sortie à votre disposition. Les paramètres doivent avoir un type de données précis.

Le principe des paramètres EN (entrée de validation) et ENO (sortie de validation) est expliqué ci-dessous. Pour plus d'informations sur les paramètres d'entrée et de sortie, reportez-vous à la description de chaque opération dans ce manuel.

Tableau 6-4 Opération CONT sous forme de pavé avec entrées et sorties

Pavé	Nom	Paragraphe dans ce manuel
	Diviser nombre à virgule flottante	12.5

Paramètres EN et ENO

L'activation de l'entrée de validation EN d'un pavé CONT provoque l'exécution de la fonction correspondante. Si cette fonction s'exécute sans erreur, la sortie de validation ENO fait circuler l'énergie le long du circuit. Ces paramètres EN et ENO sont de type de données BOOL et peuvent se trouver dans les zones de mémoire E, A, M, D ou L (voir tableaux 6-5 et 6-6).

EN et ENO fonctionnent selon les principes suivants :

- Si EN n'est pas activé (son état de signal est 0), la fonction n'est pas exécutée et ENO n'est pas activé (son état de signal est également 0).
- Si EN est activé (son état de signal est 1) et que la fonction représentée dans le pavé s'exécute sans erreur, ENO est également activé (son état de signal est également 1).
- Si EN est activé (son état de signal est 1) et qu'une erreur se produise pendant l'exécution de la fonction représentée dans le pavé, ENO n'est pas activé (son état de signal est 0).

Restrictions pour les pavés et les bobines

Vous ne pouvez pas disposer un pavé ou une bobine dans un trajet du courant qui ne débute pas à la barre d'alimentation gauche. Les opérations de comparaison constituent une exception.

Zones de mémoire et leur fonction

La plupart des opérandes en CONT se rapportent à des zones de mémoire. Le tableau ci-après présente ces zones et leur fonction.

Tableau 6-5 Zones de mémoire et leur fonction

Nom de la zone	Fonction de la zone	Accès à la zone par des éléments de la taille suivante	Abrév.
Mémoire image des entrées (MIE)	Au début du cycle, le système d'exploitation lit les entrées provenant du processus et enregistre ces valeurs dans la MIE. Le programme peut utiliser ces valeurs pendant son traitement normal.	Bit d'entrée Octet d'entrée Mot d'entrée Double mot d'entrée	E EB EW ED
Mémoire image des sorties (MIS)	Pendant le cycle, le programme calcule les valeurs de sortie et les dépose dans la MIS. A la fin du cycle, le système d'exploitation lit les valeurs de sortie calculées figurant dans cette zone et les transmet aux sorties du processus.	Bit de sortie Octet de sortie Mot de sortie Double mot de sortie	A AB AW AD
Mémentos	Cette zone permet de sauvegarder des résultats intermédiaires calculés dans le programme.	Bit de mémento Octet de mémento Mot de mémento Double mot de mémento	M MB MW MD
Périphérie d'entrée : entrées externes Périphérie de sortie : sorties externes	Cette zone permet à votre programme d'accéder directement aux modules d'entrées et de sorties, c'est-à-dire aux entrées et sorties périphériques.	Octet de périphérie d'entrée Mot de périphérie d'entrée Double mot de périphérie d'entrée Octet de périphérie de sortie Mot de périphérie de sortie Double mot de périphérie de sortie	PEB PEW PED PAB PAW PAD
Temporisations	Les temporisations sont des éléments fonctionnels de la programmation en CONT. Cette zone sert d'espace mémoire pour les cellules de temporisation. Dans cette zone, l'horloge accède aux cellules de temporisation afin de les mettre à jour en décrémentant la valeur de temps. Les opérations de temporisation accèdent aux cellules de temporisation ici.	Temporisation	T
Compteurs	Les compteurs sont des éléments fonctionnels de la programmation en CONT et cette zone leur sert d'espace mémoire. Les opérations de comptage accèdent aux compteurs ici.	Compteur	Z
Blocs de données	Cette zone contient des données accessibles à partir de tous les blocs. Si vous devez ouvrir deux blocs de données simultanément, vous pouvez utiliser l'instruction AUF DB pour l'un et l'instruction AUF DI pour l'autre. Ainsi la CPU sait auquel des deux blocs de données votre programme veut accéder pendant que ces deux blocs sont ouverts. La désignation des opérandes, L DBWi et L DIWi par exemple, détermine à quel bloc de données le programme accède. AUF DI permet certes d'ouvrir n'importe quel bloc de données, mais son but principal est l'appel des blocs de données d'instance associés aux blocs fonctionnels (FB) et aux blocs fonctionnels système (SFB). Pour plus d'informations sur les FB, les SFB et les blocs de données d'instance, consultez le manuel de programmation **/234/** et le manuel d'utilisation **/231/**.	Bloc de données ouvert avec l'instruction AUF DB : Bit de données Octet de données Mot de données Double mot de données Bloc de données ouvert avec l'instruction AUF DI : Bit de données Octet de données Mot de données Double mot de données	 DBX DBB DBW DBD DIX DIB DIW DID
Données locales	Cette zone contient des données temporaires – aussi appelées données locales dynamiques – utilisées dans des blocs de code (FB ou FC). Ces données servent de mémoire tampon intermédiaire. Une fois le bloc de code achevé, elles sont perdues. Ces données sont rangées dans la pile des données locales (pile L).	Bit de données locales temporaires Octet de données locales temporaires Mot de données locales temporaires Double mot de données locales temporaires	L LB LW LD

Le tableau 6-6 présente les plages d'adresses maximales pour les différentes zones de mémoire. Vous trouverez des informations sur les plages d'adresses propres à votre CPU dans le manuel de CPU S7-300 correspondant.

Tableau 6-6 Zones de mémoire avec leur plage d'adresses

Nom de la zone	Accès à la zone via des éléments de la taille suivante	Abrév.	Plage d'adresses maximale
Mémoire image des entrées	Bit d'entrée Octet d'entrée Mot d'entrée Double mot d'entrée	E EB EW ED	0.0 à 65 535.7 0 à 65 535 0 à 65 534 0 à 65 532
Mémoire image des sorties	Bit de sortie Octet de sortie Mot de sortie Double mot de sortie	A AB AW AD	0.0 à 65 535.7 0 à 65 535 0 à 65 534 0 à 65 532
Mémentos	Bit de mémento Octet de mémento Mot de mémento Double mot de mémento	M MB MW MD	0.0 à 255.7 0 à 255 0 à 254 0 à 252
Périphérie d'entrée : entrées externes	Octet de périphérie d'entrée Mot de périphérie d'entrée Double mot de périphérie d'entrée	PEB PEW PED	0 à 65 535 0 à 65 534 0 à 65 532
Périphérie de sortie : sorties externes	Octet de périphérie de sortie Mot de périphérie de sortie Double mot de périphérie de sortie	PAB PAW PAD	0 à 65 535 0 à 65 534 0 à 65 532
Temporisations	Temporisation	T	0 à 255
Compteurs	Compteur	Z	0 à 255
Blocs de données	Bloc de données ouvert avec l'instruction DB —(AUF) Bit de données Octet de données Mot de données Double mot de données	 DBX DBB DBW DBD	 0.0 à 65 535.7 0 à 65 535 0 à 65 534 0 à 65 532
	Bloc de données ouvert avec l'instruction DI —(AUF) Bit de données dans le DB d'instance Octet de données Mot de données Double mot de données	 DIX DIB DIW DID	 0.0 à 65 535.7 0 à 65 535 0 à 65 534 0 à 65 532
Données locales	Bit de données locales Octet de données locales Mot de données locales Double mot de données locales	L LB LW LD	0.0 à 65 535.7 0 à 65 535 0 à 65 534 0 à 65 532

6.2 Logique booléenne et tables de vérité

Flux d'énergie

Un programme CONT suit le flux d'énergie entre les barres d'alimentation lorsqu'il passe à travers les différentes entrées, sorties et autres éléments ou pavés. De nombreuses opérations CONT fonctionnent selon les principes de la logique booléenne.

Chaque opération de combinaison interroge l'état de signal d'un contact électrique afin de déterminer s'il est à 1 (activé) ou à 0 (désactivé) et fournit le résultat correspondant. L'opération sauvegarde alors ce résultat ou l'utilise dans une combinaison booléenne. Le résultat d'une combinaison s'appelle RLG (résultat logique).
Les principes de la logique booléenne sont présentés ici sur la base des contacts à fermeture et à ouverture.

Contact à fermeture

La figure 6-1 montre deux états d'un circuit à relais avec un contact entre une barre d'alimentation et une bobine. A l'état normal, ce contact est ouvert. S'il n'est pas activé, il demeure ouvert et son état de signal est 0 (inactivé). Si ce contact reste ouvert, l'énergie de la barre d'alimentation ne peut pas exciter la bobine au bout du circuit. En revanche, s'il est activé – son état devient 1 –, l'énergie circule jusqu'à la bobine.

Le circuit de gauche de la figure 6-1 montre un contact à fermeture tel qu'il est parfois représenté dans les schémas de circuits à relais. Dans cet exemple, le circuit de droite est utilisé pour montrer que le contact a été activé et qu'il est donc fermé.

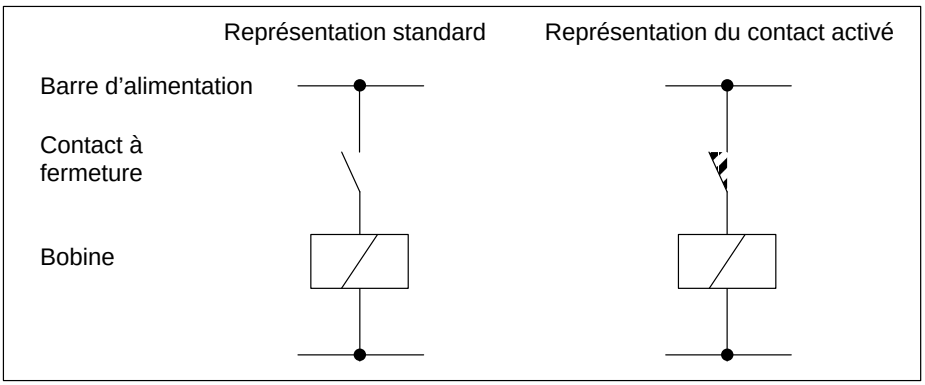


Figure 6-1 Circuit à relais avec contact à fermeture

L'opération « Contact à fermeture » (voir paragraphe 8.2) vous permet d'interroger l'état de signal d'un contact à fermeture. Elle détermine ainsi si l'énergie peut traverser le contact ou non. Si c'est le cas, l'opération fournit 1 comme résultat ; si ce n'est pas le cas, elle fournit 0 comme résultat (voir tableau 6-7). L'opération peut ensuite soit sauvegarder ce résultat, soit le combiner.

Contact à ouverture

La figure 6-2 montre deux états d'un circuit à relais avec un contact entre une barre d'alimentation et une bobine. A l'état normal, ce contact est fermé. S'il n'est pas activé, il demeure fermé et son état de signal est 0 (inactivé). Si ce contact reste fermé, l'énergie de la barre d'alimentation peut traverser le contact et exciter la bobine au bout du circuit. En revanche, s'il est activé – son état devient 1 –, le contact s'ouvre interrompant ainsi le flux d'énergie vers la bobine.

Le circuit de gauche de la figure 6-2 montre un contact à ouverture tel qu'il est parfois représenté dans les schémas de circuits à relais. Dans cet exemple, le circuit de droite est utilisé pour montrer que le contact a été activé et qu'il est donc ouvert.

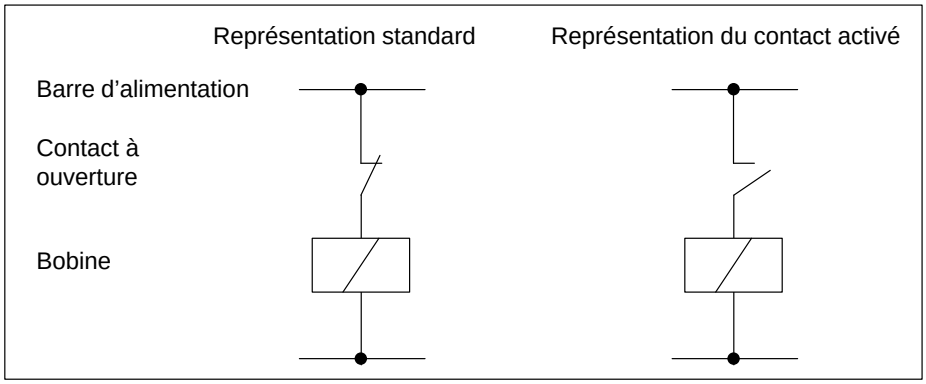


Figure 6-2 Circuit à relais avec contact à ouverture

L'opération « Contact à ouverture » (voir paragraphe 8.3) vous permet d'interroger l'état de signal d'un contact à ouverture. Elle détermine ainsi si l'énergie peut traverser le contact ou non. Si c'est le cas, l'opération fournit 1 comme résultat ; si ce n'est pas le cas, elle fournit 0 comme résultat (voir tableau 6-7). L'opération peut ensuite soit sauvegarder ce résultat, soit le combiner.

Tableau 6-7 Résultat d'une interrogation d'état de signal par « Contact à fermeture » et « Contact à ouverture »

Opération	Résultat si l'état de signal du contact est 1 (contact activé)	Résultat si l'état de signal du contact est 0 (contact désactivé)
	1 (l'énergie disponible peut circuler, car le contact à fermeture est fermé)	0 (l'énergie disponible ne peut pas circuler, car le contact à fermeture est ouvert)
	0 (l'énergie disponible ne peut pas circuler, car le contact à ouverture est ouvert)	1 (l'énergie disponible peut circuler, car le contact à ouverture est fermé)

Connexion de contacts en série

La figure 6-3 montre une séquence combinatoire d'opérations CONT qui représente deux contacts à fermeture, connectés en série à une bobine. Les contacts sont identifiés par E pour « entrée » et la bobine par A pour « sortie ». Lorsque les deux contacts de la séquence combinatoire sont activés (c'est-à-dire fermés), l'énergie peut circuler de la barre d'alimentation à travers chaque contact pour exciter la bobine au bout du circuit. Ainsi, l'énergie atteint la bobine lorsque les deux contacts E 1.0 **et** E 1.1 sont activés.

Les deux contacts sont activés dans le schéma 1. Activer un contact à fermeture ferme ce contact : l'énergie peut circuler de la barre d'alimentation à travers chaque contact fermé afin d'exciter la bobine au bout du circuit.

Dans les schémas 2 et 3, l'énergie ne peut pas circuler jusqu'à la bobine – qui n'est donc pas excitée – car l'un des deux contacts n'est pas activé.

Aucun contact n'est activé dans le schéma 4. Les deux contacts restent ouverts et l'énergie ne peut pas circuler vers la bobine qui n'est donc pas excitée.

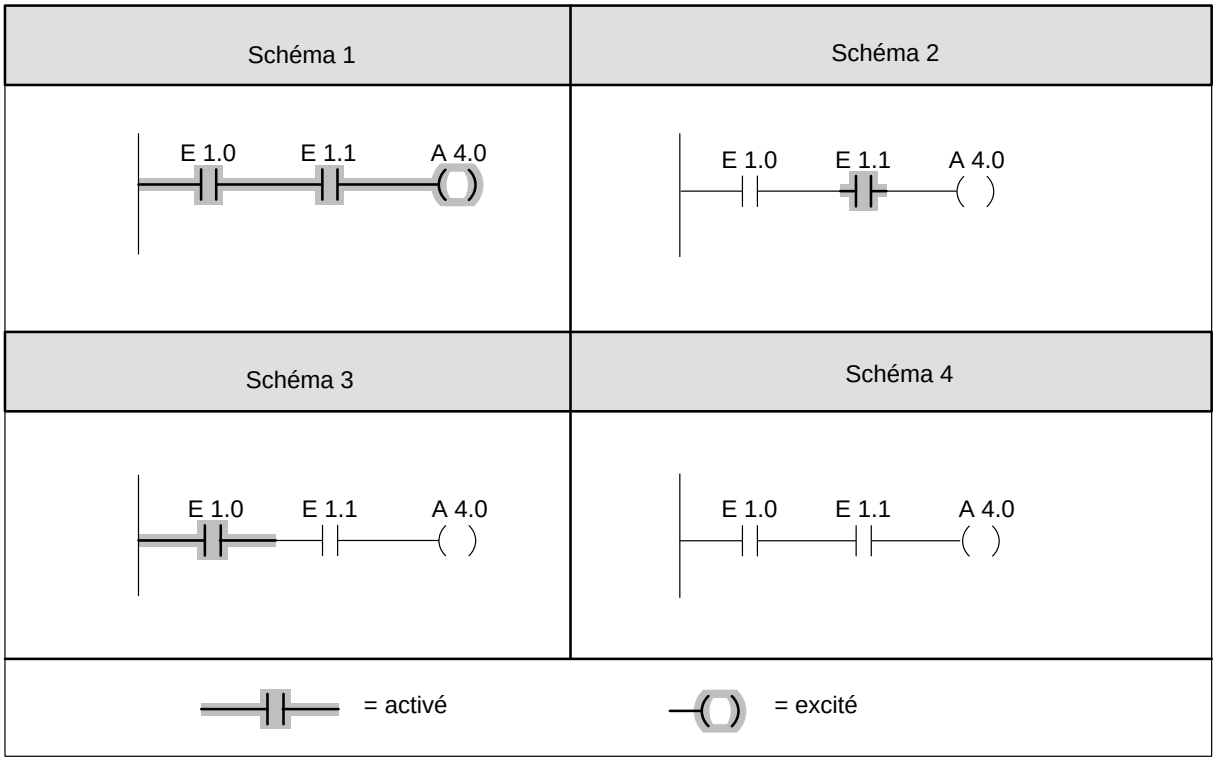


Figure 6-3 Programmation de contacts en série avec « Contact à fermeture »

Utilisation de « Contact à fermeture » en série

La figure 6-3 présente un schéma CONT avec lequel vous pouvez programmer deux contacts à fermeture connectés en série à une bobine. La première opération « Contact à fermeture » dans la séquence combinatoire interroge l'état de signal du premier contact dans la série (entrée E 1.0) et fournit le résultat 0 ou 1 en conséquence (voir tableau 6-7). 1 comme résultat signifie que le contact est fermé et que l'énergie disponible peut traverser le contact. 0 comme résultat signifie que le contact est ouvert, interrompant le flux d'énergie disponible au contact.
La première opération « Contact à fermeture » copie ce 1 ou ce 0 dans le bit RLG (résultat logique) du mot d'état de l'automate programmable.

La seconde opération « Contact à fermeture » dans la séquence combinatoire interroge l'état de signal du second contact dans la série (E 1.1) et fournit un résultat égal à 1 ou à 0 selon que ce contact est ouvert ou fermé (voir tableau 6-7). Puis, elle combine le résultat de l'interrogation de l'état de signal du second contact à la valeur figurant dans le bit RLG. Le résultat de cette combinaison – 0 ou 1 – remplace l'ancienne valeur dans le bit RLG du mot d'état. L'opération « Sortie » (voir paragraphe 8.4) affecte cette nouvelle valeur à la bobine (sortie A 4.0).

On peut représenter les résultats possibles d'une telle combinaison dans une table de vérité, 1 signifiant « vrai » et 0 « faux ». Les combinaisons possibles et leurs résultats sont résumés dans le tableau 6-8 ; « contact fermé » et « flux d'énergie » correspondent à « vrai » et « contact ouvert » et « pas de flux d'énergie » correspondent à « faux » (voir figure 6-3 pour les contacts).

Tableau 6-8 Table de vérité ET

Si le résultat fourni par l'interrogation de l'état de signal du contact E 1.0 est	**et que le résultat fourni par l'interrogation de l'état de signal du contact E 1.1 est**	**le résultat de la combinaison présentée à la figure 6-3 est**
1 (contact fermé)	1 (contact fermé)	1 (flux d'énergie)
0 (contact ouvert)	1 (contact fermé)	0 (pas de flux d'énergie)
1 (contact fermé)	0 (contact ouvert)	0 (pas de flux d'énergie)
0 (contact ouvert)	0 (contact ouvert)	0 (pas de flux d'énergie)

Connexion de contacts en parallèle

La figure 6-4 montre une séquence combinatoire d'opérations CONT dans laquelle deux contacts à fermeture sont connectés en parallèle à une bobine. Les contacts sont identifiés par E pour « entrée » et la bobine par A pour « sortie ». Activer un contact à fermeture ferme ce contact. Si **l'un** des deux contacts de la séquence combinatoire est activé (c'est-à-dire fermé), l'énergie peut circuler via E 1.0 ou E 1.1 de la barre d'alimentation jusqu'à la bobine au bout du circuit et l'exciter. Si les deux contacts sont fermés, l'énergie parvient également à la bobine qu'elle excite.

Dans les schémas 1 et 2, un contact est activé et l'autre pas. Activer un contact à fermeture ferme ce contact : l'énergie peut circuler de la barre d'alimentation à travers le contact fermé jusqu'à la bobine au bout du circuit. Comme les deux contacts sont connectés en parallèle, il suffit que l'un d'eux soit fermé pour que l'énergie parvienne à la bobine au bout du circuit et l'excite.

Dans le schéma 3, les deux contacts sont activés : l'énergie peut traverser les deux contacts fermés et exciter la bobine au bout du circuit.

Aucun contact n'est activé dans le schéma 4. Ils restent tous deux ouverts et l'énergie ne peut pas circuler vers la bobine qui n'est donc pas excitée.

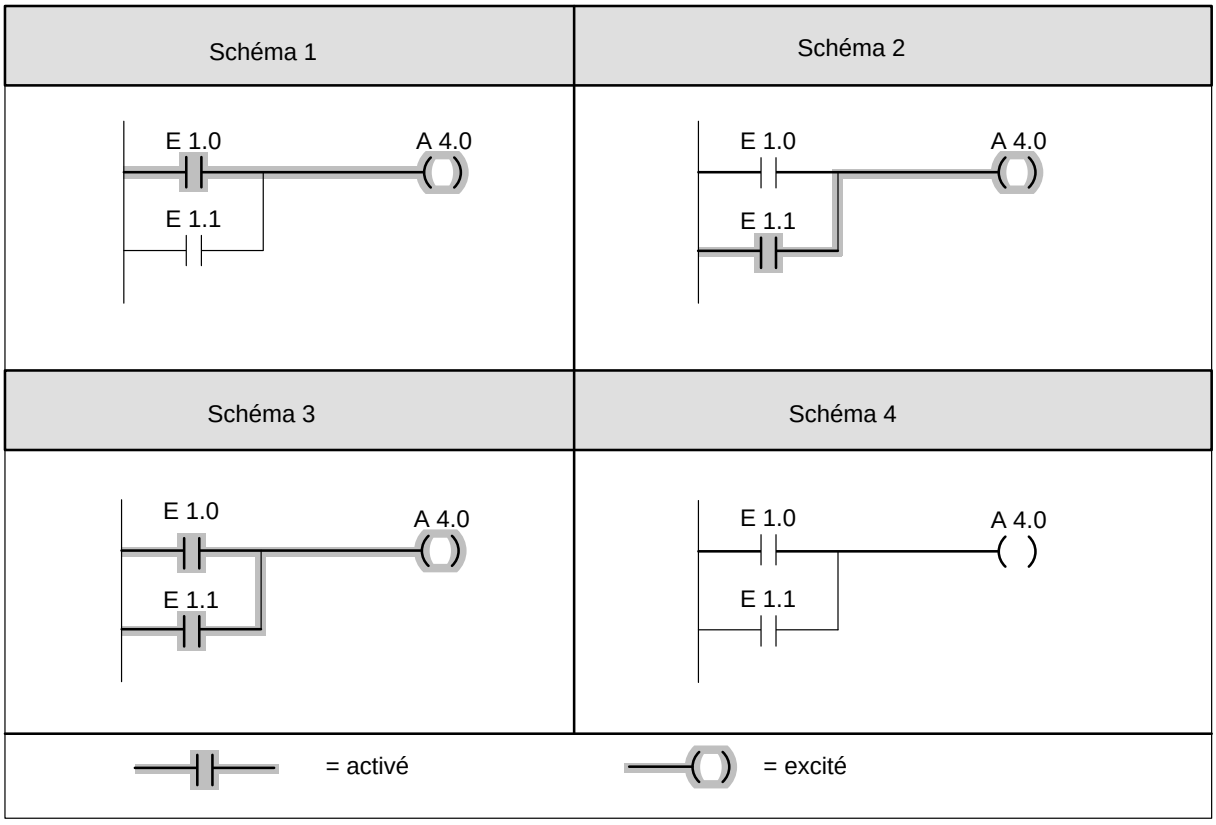


Figure 6-4 Programmation de contacts en parallèle avec « Contact à fermeture »

Utilisation de « Contact à fermeture » en parallèle

La figure 6-4 présente un schéma CONT avec lequel vous pouvez programmer deux contacts à fermeture connectés en parallèle à une bobine. La première opération « Contact à fermeture » dans la séquence combinatoire interroge l'état de signal du premier contact (entrée E 1.0) et fournit le résultat 0 ou 1 en conséquence (voir tableau 6-7). 1 comme résultat signifie que le contact est fermé et que l'énergie disponible peut traverser le contact. 0 comme résultat signifie que le contact est ouvert, interrompant le flux d'énergie disponible au contact. La première opération « Contact à fermeture » copie ce 1 ou ce 0 dans le bit RLG (résultat logique) du mot d'état de l'automate programmable.

La seconde opération « Contact à fermeture » dans la séquence combinatoire interroge l'état de signal du second contact (E 1.1) et fournit un résultat égal à 1 ou à 0 selon que ce contact est ouvert ou fermé (voir tableau 6-7). Puis, elle combine le résultat de l'interrogation de l'état de signal du second contact à la valeur figurant dans le bit RLG. Le résultat de cette combinaison – 0 ou 1 – remplace l'ancienne valeur dans le bit RLG du mot d'état. L'opération « Sortie » (voir paragraphe 8.4) affecte cette nouvelle valeur à la bobine (sortie A 4.0).

On peut représenter les résultats possibles d'une telle combinaison dans une table de vérité, 1 signifiant « vrai » et 0 « faux ». Les combinaisons possibles et leurs résultats sont résumés dans le tableau 6-9 ; « contact fermé » et « flux d'énergie » correspondent à « vrai » et « contact ouvert » et « pas de flux d'énergie » correspondent à « faux » (voir figure 6-4 pour les contacts).

Tableau 6-9 Table de vérité OU

Si le résultat fourni par l'interrogation de l'état de signal du contact E 1.0 est	et que le résultat fourni par l'interrogation de l'état de signal du contact E 1.1 est	le résultat de la combinaison présentée à la figure 6-4 est
1 (contact fermé)	0 (contact ouvert)	1 (flux d'énergie)
0 (contact ouvert)	1 (contact fermé)	1 (flux d'énergie)
1 (contact fermé)	1 (contact fermé)	1 (flux d'énergie)
0 (contact ouvert)	0 (contact ouvert)	0 (pas de flux d'énergie)

6.3 Signification des registres de la CPU dans les instructions

Explication

Les registres assistent la CPU dans l'exécution d'opérations de combinaison, d'opérations arithmétiques, d'opérations de décalage ou de conversion. Ces registres sont décrits ci-après.

Accumulateurs

Les deux accumulateurs de 32 bits sont des registres à usage général permettant de traiter octets, mots et doubles mots.

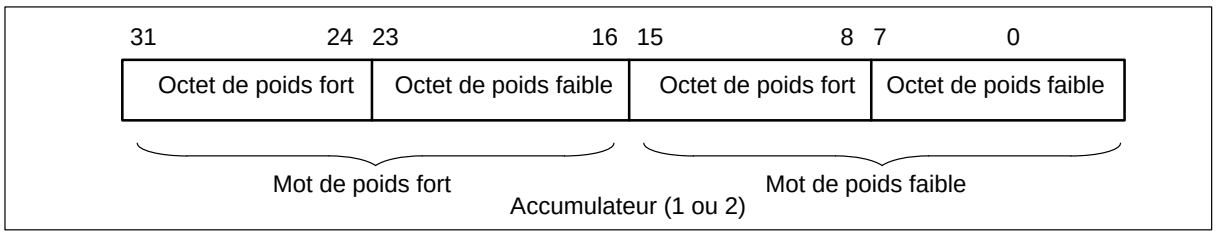


Figure 6-5 Zones d'un accumulateur

Mot d'état

Le mot d'état contient des bits auxquels vous pouvez accéder dans l'opérande des combinaisons sur bits. Les paragraphes suivant cette figure décrivent les bits 0 à 8.



Figure 6-6 Organisation du mot d'état

Modification des bits dans le mot d'état

Valeur	Signification
0	met l'état de signal à 0
1	met l'état de signal à 1
x	modifie l'état
–	état reste inchangé

Première interrogation

Le bit 0 du mot d'état est appelé première interrogation (bit /PI ; voir figure 6-6). Au début d'un réseau CONT, l'état de signal du bit /PI est toujours 0 à moins que la séquence ne se termine par —(SAVE). La barre oblique devant /PI indique qu'il s'agit d'une inversion, c'est-à-dire que la valeur est toujours 0 au début d'un réseau CONT.

Chaque opération combinatoire interroge l'état de signal du bit /PI et du contact désigné. L'état de signal du bit /PI détermine l'exécution d'une séquence combinatoire. Si le bit /PI est égal à 0 (au début d'un réseau CONT), l'opération va sauvegarder le résultat dans le bit du résultat logique du mot d'état et met le bit /PI à 1. Cette procédure d'interrogation s'appelle première interrogation. La valeur 1 ou 0 mémorisée dans le bit RLG après la première interrogation est ensuite appelée résultat de la première interrogation.

Lorsque le bit /PI est égal à 1, l'opération combine le résultat de l'interrogation de l'état de signal du contact désigné avec la valeur mémorisée dans le bit RLG précédent et sauvegarde le résultat dans le bit RLG.

Un trajet de courant d'opérations CONT (séquence d'opérations combinatoires) se termine toujours par une opération de sortie (« Mettre à 1 », « Mettre à 0 » ou « Sortie ») ou par une opération de saut relative au résultat logique. Ces opérations remettent le bit /PI à 0.

Résultat logique

Le bit 1 du mot d'état est appelé résultat logique (bit RLG ; voir figure 6-6). Il contient le résultat d'une séquence d'opérations combinatoires sur bits ou d'opérations de comparaison. L'état de signal du RLG peut fournir des informations sur le flux d'énergie.

La première opération dans un réseau CONT interroge l'état de signal d'un contact et obtient le résultat 0 ou 1. L'opération mémorise le résultat de l'interrogation de l'état de signal dans le bit RLG. La deuxième opération de la séquence combinatoire interroge également l'état de signal d'un contact et donne un résultat. L'opération combine ensuite ce résultat avec la valeur figurant dans le mot d'état selon les règles de la logique booléenne (voir « Première interrogation » ci-avant et le chapitre 8). Le résultat de cette combinaison est sauvegardé dans le bit RLG du mot d'état où il se substitue à l'ancienne valeur du RLG. Chaque opération suivante dans la séquence combinatoire exécute une combinaison de deux valeurs : le résultat obtenu lors de l'interrogation de l'état de signal du contact et le RLG en cours.

Vous pouvez, par exemple, affecter à l'aide d'une opération booléenne l'état d'un mémento booléen au RLG lors d'une première interrogation, ou bien, réaliser des opérations de saut.

Bit d'état

Le bit 2 du mot d'état est appelé bit d'état (bit ETAT, voir figure 6-6). Le bit d'état contient la valeur d'un bit en accès. L'état d'une opération de combinaison ayant accès en lecture à la mémoire (Contact à fermeture, Contact à ouverture) est toujours le même que la valeur du bit interrogé par cette opération (bit sur lequel elle effectue sa combinaison). L'état d'une opération de combinaison ayant accès en écriture à la mémoire (Mettre à 1, Mettre à 0 ou Sortie) est identique à la valeur du bit dans lequel l'opération écrit ou bien, si aucune écriture n'a lieu, à la valeur du bit auquel accède l'opération. Le bit d'état n'a pas de signification pour les opérations de combinaison qui n'accèdent pas à la mémoire ; ces opérations mettent le bit d'état à 1. Le bit d'état n'est pas interrogé par des opérations, mais uniquement exploité pendant le test du programme (état du programme).

Bit OU

Le bit 3 du mot d'état est appelé bit OU (voir figure 6-6). Le bit OU est nécessaire lorsque vous exécutez une combinaison ET avant une combinaison OU à l'aide d'opérations de contact. Une combinaison OU équivaut au branchement en parallèle des contacts. Une combinaison ET correspond au branchement en série des contacts (voir paragraphe 6.2). Une combinaison ET peut contenir les opérations « Contact à fermeture », « Contact à ouverture ». Le bit OU indique à ces opérations qu'une combinaison ET exécutée précédemment a fourni la valeur 1, anticipant ainsi le résultat de la combinaison OU. Toute autre opération traitant des bits remet le bit OU à 0.

Bit de débordement

Le bit 5 du mot d'état est appelé bit de débordement (bit DEB, voir figure 6-6). Le bit DEB est mis à 1 par une opération arithmétique ou une opération de comparaison de nombres à virgule flottante lorsqu'apparaît une erreur telle que débordement, opération illicite, nombre à virgule flottante illicite. Ce bit est mis à 0 selon le résultat de l'opération.

Bit de débordement mémorisé

Le bit 4 du mot d'état est appelé bit de débordement mémorisé (bit DM, voir figure 6-6). Le bit DM est mis à 1 en même temps que le bit DEB lorsqu'une erreur se produit. DM restant inchangé une fois les opérations arithmétiques exécutées correctement (contrairement au bit de débordement), il indique si une erreur s'est produite dans l'une des opérations exécutées précédemment. Les opérations suivantes remettent le bit DM à 0 : SPS (saut si DM = 1, programmation LIST), appels de bloc et fin de bloc.

Bits indicateurs BI1 et BI0

Les bits 7 et 6 du mot d'état sont appelés bit indicateur 1 et bit indicateur 0 (BI1 et BI0, voir figure 6-6). Les bits indicateurs BI1 et BI0 donnent des informations sur les résultats ou bits suivants :

- résultat d'une opération arithmétique,
- résultat d'une opération de comparaison,
- résultat d'une opération combinatoire sur mots,
- bits décalés par une opération de rotation ou de décalage à partir d'un opérande.

Les tableaux 6-10 à 6-15 présentent la signification de BI1 et BI0 après l'exécution de certaines opérations par votre programme.

Tableau 6-10 BI1 et BI0 après des opérations arithmétiques (sans débordement)

BI1	BI0	Explication
0	0	Résultat = 0
0	1	Résultat < 0
1	0	Résultat > 0

Tableau 6-11 BI1 et BI0 après des opérations arithmétiques sur des entiers (avec débordement)

BI1	BI0	Explication
0	0	Débordement de plage négatif pour additionner des entiers de 16 bits et additionner des entiers de 32 bits
0	1	Débordement de plage négatif pour multiplier des entiers de 16 bits et multiplier des entiers de 32 bits Débordement de plage positif pour additionner des entiers de 16 bits, soustraire des entiers de 16 bits, additionner des entiers de 32 bits, soustraire des entiers de 32 bits, complémenter à 2 des entiers de 16 bits et complémenter à 2 des entiers de 32 bits
1	0	Débordement de plage positif pour multiplier des entiers de 16 bits, multiplier des entiers de 32 bits, diviser des entiers de 16 bits et diviser des entiers de 32 bits Débordement de plage négatif pour additionner des entiers de 16 bits, soustraire des entiers de 16 bits, additionner des entiers de 32 bits et soustraire des entiers de 32 bits
1	1	Division par 0 pour diviser des entiers de 16 bits, diviser des entiers de 32 bits et reste de division (32 bits)

Tableau 6-12 BI1 et BI0 après des opérations arithmétiques sur des nombres à virgule flottante (avec débordement)

BI1	BI0	Explication
0	0	Débordement bas graduel
0	1	Débordement de plage négatif
1	0	Débordement de plage positif
1	1	Opération illicite

Tableau 6-13 BI1 et BI0 après des opérations de comparaison

BI1	BI0	Explication
0	0	IN2 = IN1
0	1	IN2 < IN1
1	0	IN2 > IN1
1	1	Ni IN1 ni IN2 n'est un nombre à virgule flottante valable

Tableau 6-14 BI1 et BI0 après des opérations de rotation et de décalage

BI1	BI0	Explication
0	0	Dernier bit décalé = 0
1	0	Dernier bit décalé = 1

Tableau 6-15 BI1 et BI0 après des opérations combinatoires sur mots

BI1	BI0	Explication
0	0	Résultat = 0
1	0	Résultat <> 0

Résultat binaire

Le bit 8 du mot d'état est appelé bit de résultat binaire (bit RB, voir figure 6-6). Le bit RB constitue un lien entre le traitement de bits et de mots. Il permet à votre programme d'exploiter le résultat d'une opération combinatoire sur mots comme un résultat binaire et d'intégrer ce résultat à une séquence de combinaison sur bits. Ainsi, le résultat binaire (RB) s'assimile à un mémento interne permettant de protéger le résultat logique (RLG) contre toute modification apportée par une opération sur mots, et de le retrouver après l'opération destinée à reprendre la séquence combinatoire sur bits interrompue.

Le bit RB vous permet, par exemple, d'écrire un bloc fonctionnel (FB) ou une fonction (FC) en liste d'instructions, puis d'appeler ce FB ou cette FC en schéma à contacts CONT.

Lorsque vous écrivez un bloc fonctionnel ou une fonction que vous voulez appeler en CONT, que vous le fassiez en LIST ou en CONT, il faudra de toute manière tenir compte du bit RB. Celui-ci correspond à la sortie de validation (ENO) d'un pavé CONT. Pour mémoriser le RLG dans le bit RB, vous pouvez utiliser l'opération SAVE (en LIST) ou la bobine —(SAVE) (en CONT), en tenant compte des critères suivants :

- Mémorisez un RLG égal à 1 dans le bit RB au cas où le FB ou la FC sont exécutés sans erreur.
- Mémorisez un RLG égal à 0 dans le bit RB au cas où le FB ou la FC sont exécutés avec erreur.

Vous devez programmer ces instructions à la fin du FB ou de la FC afin qu'elles soient les dernières opérations exécutées dans le bloc.



Attention

Le bit RB peut être remis à 0 involontairement.

Lorsque vous écrivez des FB et FC en CONT sans utiliser le bit RB comme décrit précédemment, un FB ou une FC est susceptible d'écraser le bit RB d'un autre FB ou d'une autre FC.

Pour éviter cette erreur, vous devez mémoriser le RLG à la fin de chaque FB ou FC, comme décrit ci-avant.

Signification EN/ENO

Les paramètres de l'entrée de validation (EN) et de la sortie de validation (ENO) d'un pavé CONT suivent les principes suivants :

- Si EN n'est pas activée (c'est-à-dire si son état de signal est égal à 0), le pavé n'exécute pas sa fonction et ENO n'est donc pas activée (c'est-à-dire son état de signal est aussi égal à 0).
- Si EN est activée (c'est-à-dire si son état de signal est égal à 1) et si le pavé correspondant exécute sa fonction sans erreur, alors ENO est également activée (c'est-à-dire son état de signal est aussi égal à 1).
- Si EN est activée (c'est-à-dire si son état de signal est égal à 1) et si une erreur se produit durant l'exécution de la fonction du pavé correspondant, alors ENO n'est pas activée (c'est-à-dire son état de signal est égal à 0).

Si dans votre programme vous appelez un bloc fonctionnel système (SFB) ou une fonction système (SFC), le SFB ou la SFC indique par l'état de signal du bit RB si la CPU a exécuté la fonction sans erreur ou avec erreur :

- Si une erreur s'est produite durant l'exécution de la fonction, le bit RB est égal à 0.
- Si aucune erreur ne s'est produite durant l'exécution de la fonction, le bit RB est égal à 1.

Adressage

7

Contenu de ce chapitre

Paragraphe	Thème	Page
7.1	Présentation	7-2
7.2	Types d’opérandes	7-4

7.1 Présentation

Qu'est-ce que l'adressage ?

De nombreuses opérations CONT utilisent un ou plusieurs opérandes qui indiquent une constante ou une adresse où figure la valeur sur laquelle doit porter l'opération. L'adresse d'opérande peut représenter un bit, un octet, un mot ou un double mot.

L'opérande peut désigner l'un des éléments suivants :

- une constante, la valeur d'une temporisation ou d'un compteur ou une chaîne de caractères ASCII,
- une adresse dans le mot d'état de l'automate programmable,
- un bloc de données et une adresse dans la plage du bloc de données,

Adressage immédiat et adressage direct

Vous disposez des deux types d'adressage suivants :

- l'adressage immédiat (indication d'une constante comme opérande)
- l'adressage direct (indication d'une variable comme opérande)

La figure 7-1 représente un exemple d'adressage immédiat et d'adressage direct. La fonction du pavé est de comparer deux paramètres d'entrée (dans le cas présent, deux nombres entiers de 16 bits) pour savoir si le premier est inférieur ou égal au second. La constante 50 est entrée comme paramètre d'entrée IN1 et le mot de mémento MW200, une adresse de mémoire, comme paramètre d'entrée IN2.

Puisque dans l'exemple considéré, 50 est la valeur effective que IN1 va utiliser, 50 est considéré comme l'opérande immédiat du pavé. Etant donné que MW200 pointe sur une adresse en mémoire où se trouve une autre valeur qui va être utilisée par IN2, MW200 est considéré comme opérande direct. En fait, MW200 est une adresse, et non pas la valeur elle-même.

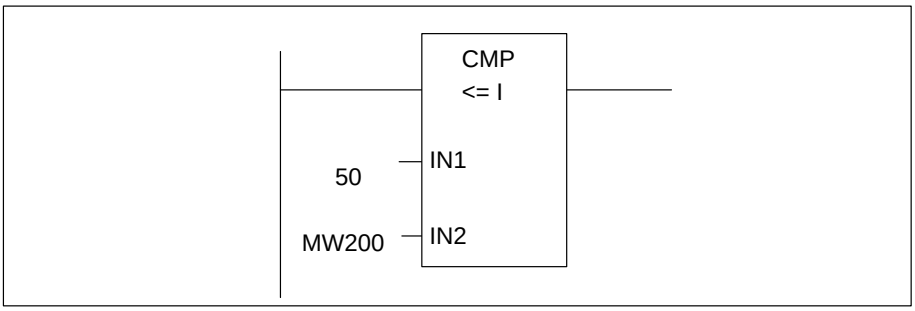


Figure 7-1 Adressage immédiat et adressage direct

Tableau 7-1 Formats constants pour l'adressage immédiat avec des opérandes de type de données élémentaire

Type et description	Taille en bits	Formats	Plage et représentation des nombres (valeur inférieure à valeur supérieure)	Exemple
BOOL (bit)	1	Texte booléen	TRUE/FALSE	TRUE
BYTE (octet)	8	Nombre hexadécimal	B#16#0 à B#16#FF	B#16#10 byte#16#10
WORD (mot)	16	Nombre binaire Nombre hexadécimal Nombre DCB Nombre décimal non signé	2#0 à 2#1111_1111_1111_1111 W#16#0 à W#16#FFFF C#0 à C#999 B#(0,0) à B#(255,255)	2#0001_0000_0000_0000 W#16#1000 word16#1000 C#998 B#(10,20) byte#(10,20)
DWORD (double mot)	32	Nombre binaire Nombre hexadécimal Nombre décimal non signé	2#0 à 2#1111_1111_1111_1111_ 1111_1111_1111_1111 DW#16#0000_0000 à DW#d16#FFFF_FFFF B#(0,0,0,0) à B#(255,255,255,255)	2#1000_0001_0001_1000_ 1011_1011_0111_1111 DW#16#00A2_1234 dword#16#00A2_1234 B#(1, 14, 100, 120) byte#(1,14,100,120)
INT (entier de 16 bits)	16	Nombre décimal signé	-32768 à 32767	1
DINT (entier de 32 bits ou double)	32	Nombre décimal signé	L#–2147483648 à L#2147483647	L#1
REAL (virgule flottante)	32	Nombre à virgule flottante IEEE	Limite supérieure : ±3.402823e+38 Limite inférieure : ±1.175 495e-38 (voir aussi tableau C-5)	1,234567e+13
S5TIME (durée SIMATIC)	16	Durée S5 en unités de 10 ms (présélection)	S5T#0H_0M_0S_10MS à S5T#2H_46M_30S_0MS et S5T#0H_0M_0S_0MS	S5T#0H_1M_0S_0MS S5Time#0H_1H_1M_0S_0MS
TIME (durée CEI)	32	Durée CEI en unités de 1 ms, nombre entier signé	T#-24D_20H_31M_23S_648MS à T#24D_20H_31M_23S_647MS	T#0D_1H_1M_0S_0MS TIME#0D_1H_1M_0S_0MS
DATE (date CEI)	16	Date CEI en unités de 1 jour	D#1990-1-1 à D#2168-12-31	D#1994-3-15 DATE#1994–3–15
TIME_OF_ DAY (heure du jour)	32	Heure du jour en unités de 1 ms	TOD#0:0:0.0 à TOD #23:59:59.999	TOD#1:10:3.3 TIME_OF_DAY#1:10:3.3
CHAR (caractère)	8	Caractère ASCII	'A','B' etc.	'E'

7.2 Types d'opérandes

Opérandes possibles

L'opérande d'une opération CONT peut désigner l'un des éléments suivants :

- un bit dont l'état de signal doit être interrogé,
- un bit auquel est affecté l'état de signal de la séquence combinatoire,
- un bit auquel est affecté le résultat logique RLG,
- un bit qui doit être mis à un ou à zéro,
- un nombre indiquant le compteur à incrémenter ou à décrémenter,
- un nombre indiquant la temporisation à utiliser,
- un mémento de front qui mémorise le RLG précédent,
- un mémento de front qui mémorise l'état de signal précédent d'un autre opérande,
- un octet, mot ou double mot qui contient la valeur qui va être utilisée par l'élément CONT ou le pavé CONT,
- le numéro d'un bloc de données DB ou d'un bloc de données d'instance DI à ouvrir ou à créer,
- le numéro d'une fonction (FC), d'une fonction système (SFC), d'un bloc fonctionnel (FB) ou d'un bloc fonctionnel système (SFB) à appeler,
- un repère de saut auquel sauter.

Identificateurs d'opérandes

Les opérandes comme variables sont constitués d'un identificateur d'opérande et d'une adresse à l'intérieur de la zone de mémoire indiquée par l'identificateur d'opérande. Un identificateur d'opérande peut appartenir à l'un des deux types ci-après.

- Un identificateur d'opérande qui indique les deux objets de données suivants :
  - zone de mémoire où se trouve la valeur (objet de données) sur laquelle doit porter l'opération (par exemple, E pour mémoire image des entrées ; voir tableau 6-5)
  - taille de la valeur (objet de données) sur laquelle doit porter l'opération (par exemple, B pour octet, W pour mot, D pour double mot ; voir tableau 6-5)
- Un identificateur d'opérande qui indique une zone de mémoire mais pas la taille de l'objet de données dans cette zone (par exemple, un identificateur pour la zone T (temporisation), Z (compteur) ou DB ou DI (bloc de données), plus le numéro de la temporisation, du compteur ou du bloc de données ; voir tableau 6-5)

Pointeur

Un pointeur permet d'identifier l'adresse d'une variable. Il contient un opérande au lieu d'une valeur. Lorsque vous affectez un paramètre effectif au type de paramètre « pointeur », vous indiquez l'adresse en mémoire. Dans STEP 7, vous pouvez indiquer le pointeur en format pointeur ou simplement comme opérande (p. ex. M50.0). Dans l'exemple suivant, le format « pointeur » permet d'adresser des données à partir de M 50.0 :

P#M50.0

Utilisation de mots ou doubles mots comme objets de données

Lorsque vous utilisez une opération dont l'identificateur d'opérande indique une zone de mémoire dans votre automate programmable, ainsi qu'un objet de données de la taille d'un mot ou d'un double mot, vous devez tenir compte du fait que l'adresse en mémoire est toujours référencée comme adresse d'octet. Celle-ci correspond au numéro de l'octet de poids faible ou au numéro de l'octet de poids fort dans le mot ou double mot. L'opérande de l'instruction dans l'exemple de la figure 7-2 adresse quatre octets successifs dans la zone de mémoire M, à partir de l'octet 10 (MB10) et jusqu'à l'octet 13 (MB13).

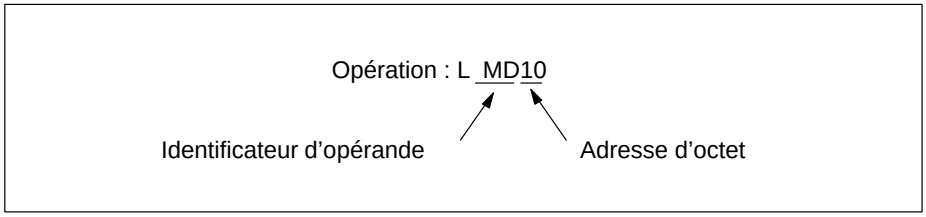


Figure 7-2 Exemple d'adresse en mémoire référencée comme adresse d'octet

La figure 7-3 représente des objets de données avec les tailles suivantes :

- Double mot : double mot de mémento MD10
- Mot : mots de mémento MW10, MW11 et MW12
- Octet : octets de mémento MB10, MB11, MB12 et MB13

Si vous utilisez des opérandes absolus de la taille d'un mot ou d'un double mot, assurez-vous de ne pas créer d'assignations d'octets qui se chevauchent.

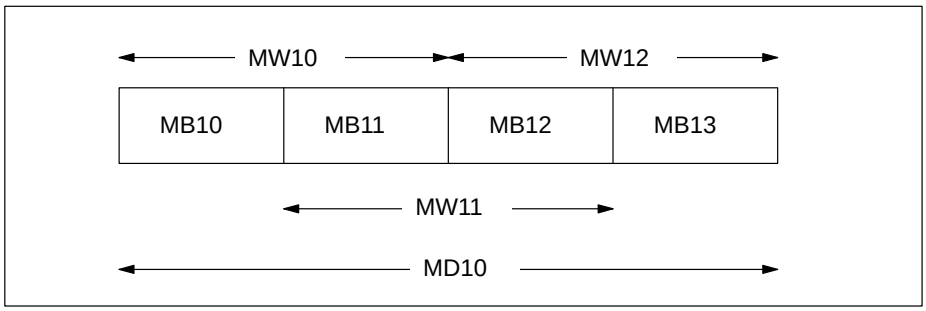


Figure 7-3 Référence à une adresse en mémoire sous forme d'adresse d'octet

Opérations combinatoires sur bits

8

Contenu de ce chapitre

Paragraphe	Thème	Page
8.1	Présentation	8-2
8.2	Contact à fermeture	8-3
8.3	Contact à ouverture	8-4
8.4	Sortie	8-5
8.5	Connecteur	8-6
8.6	Inverser RLG	8-7
8.7	Sauvegarder RLG dans RB	8-8
8.8	Mettre à 1	8-9
8.9	Mettre à 0	8-10
8.10	Initialiser compteur	8-11
8.11	Incrémenter (bobine)	8-12
8.12	Décrémenter (bobine)	8-13
8.13	Temporisation sous forme d’impulsion	8-14
8.14	Temporisation sous forme d’impulsion prolongée	8-15
8.15	Temporisation sous forme de retard à la montée	8-16
8.16	Temporisation sous forme de retard à la montée mémorisé	8-17
8.17	Temporisation sous forme de retard à la retombée	8-18
8.18	Détecter front montant du RLG	8-19
8.19	Détecter front descendant du RLG	8-20
8.20	Détecter front montant de signal	8-21
8.21	Détecter front descendant de signal	8-22
8.22	Bascule mise à 1, mise à 0	8-23
8.23	Bascule mise à 0, mise à 1	8-24

8.1 Présentation

Explication

Les opérations combinatoires sur bits utilisent deux chiffres : 1 et 0. Ces deux chiffres sont à la base du système de numération binaire et sont appelés chiffres binaires ou bits. Pour les contacts et les bobines, 1 signifie activé ou excité et 0 signifie désactivé ou désexcité.

Les opérations de combinaison sur bits évaluent les états de signal 1 et 0 et les combinent selon la logique booléenne. Le résultat de ces combinaisons est égal à 1 ou 0. Il s'agit du résultat logique (RLG, voir paragraphe 6.3). Les opérations combinatoires déclenchées par les opérations combinatoires sur bits exécutent diverses fonctions.

Fonctions

Il existe des opérations combinatoires sur bits pour effectuer les fonctions suivantes :

- « Contact à fermeture » et « Contact à ouverture » interrogent chacune l'état de signal d'un contact et leur résultat est soit copié dans le bit de résultat logique RLG, soit combiné au RLG. Si ces contacts sont connectés en série, ces opérations combinent le résultat de leur interrogation d'état de signal selon la table de vérité ET (voir tableau 6-8) ; s'ils sont connectés en parallèle, elles le combinent selon la table de vérité OU (voir tableau 6-9).
- « Sortie » et « Connecteur » assignent le RLG ou le mémorisent temporairement.
- Les opérations suivantes réagissent à un RLG égal à 1 :
  - « Mettre à 1 » et « Mettre à 0 »
  - « Bascule mise à 1, mise à 0 » et « Bascule mise à 0, mise à 1 »
- D'autres opérations exécutent les fonctions suivantes en cas de front montant ou descendant :
  - Incrémenter ou décrémenter la valeur d'un compteur
  - Démarrer une temporisation
  - Fournir une sortie égale à 1
- Les opérations restantes agissent directement sur le RLG de la manière suivante :
  - Inverser le RLG
  - Sauvegarder le RLG dans le bit de résultat binaire RB du mot d'état

Dans ce chapitre, les compteurs et temporisations sont représentés dans le format international (anglais) et le format SIMATIC (allemand).

8.2 Contact à fermeture

Description

L'opération « Contact à fermeture » vous permet d'interroger l'état de signal du contact indiqué en opérande.
Si l'état de signal est 1 à cet opérande, le contact est fermé et l'opération fournit un résultat égal à 1. En revanche, si l'état de signal est 0, le contact est ouvert et l'opération fournit un résultat égal à 0.

Lorsque « Contact à fermeture » est la première opération dans une séquence combinatoire, elle range le résultat de son interrogation d'état de signal dans le bit de résultat logique (RLG).

Lorsqu'elle n'est pas la première opération dans une séquence combinatoire, elle combine le résultat de son interrogation d'état de signal à la valeur figurant dans le bit RLG. Cette combinaison se fait de l'une des deux façons suivantes :

- S'il s'agit d'une connexion en série, l'opération combine son résultat selon la table de vérité ET.
- S'il s'agit d'une connexion en parallèle, l'opération combine son résultat selon la table de vérité OU.

Tableau 8-1 Contact à fermeture : représentation et paramètre

Elément CONT	Paramètre	Type de données	Zone de mémoire	Description
<opérande>	<opérande>	BOOL TIMER COUNTER	E, A, M, T, Z, D, L	L'opérande indique le bit dont l'état de signal est interrogé.

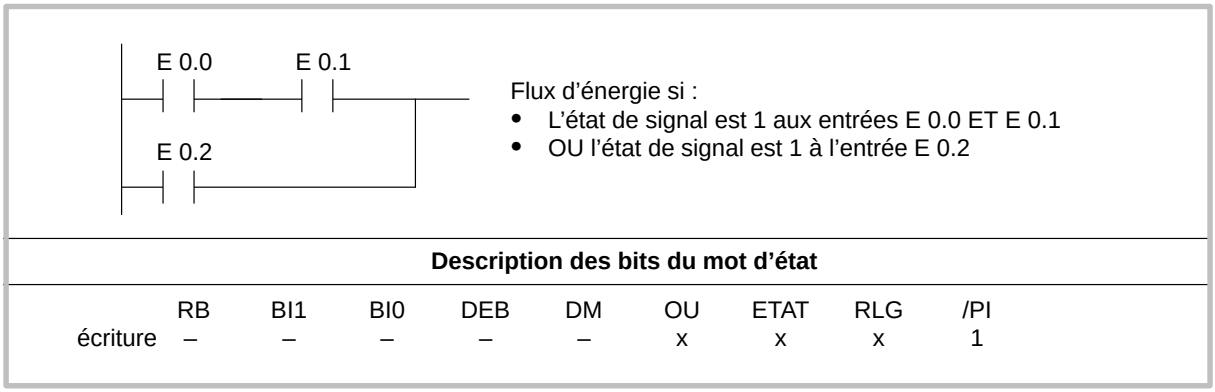

E 0.0
E 0.1
E 0.2
Flux d'énergie si :
• L'état de signal est 1 aux entrées E 0.0 ET E 0.1
• OU l'état de signal est 1 à l'entrée E 0.2
Description des bits du mot d'état
RB BI1 BI0 DEB DM OU ETAT RLG /PI
écriture – – – – – x x x 1

Figure 8-1 Contact à fermeture

8.3 Contact à ouverture

Description

L'opération « Contact à ouverture » vous permet d'interroger l'état de signal du contact indiqué en opérande.
Si l'état de signal est 0 à cet opérande, le contact est fermé et l'opération fournit un résultat égal à 1. En revanche, si l'état de signal est 1, le contact est ouvert et l'opération fournit un résultat égal à 0.

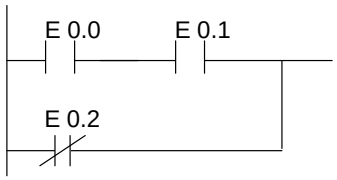
Lorsque « Contact à ouverture » est la première opération dans une séquence combinatoire, elle range le résultat de son interrogation d'état de signal dans le bit de résultat logique (RLG).

Lorsqu'elle n'est pas la première opération dans une séquence combinatoire, elle combine le résultat de son interrogation d'état de signal à la valeur figurant dans le bit RLG. Cette combinaison se fait de l'une des deux façons suivantes :

- S'il s'agit d'une connexion en série, l'opération combine son résultat selon la table de vérité ET.
- S'il s'agit d'une connexion en parallèle, l'opération combine son résultat selon la table de vérité OU.

Tableau 8-2 Contact à ouverture : représentation et paramètre

Elément CONT	Paramètre	Type de données	Zone de mémoire	Description
<opérande> 	<opérande>	BOOL TIMER COUNTER	E, A, M, T, Z, D, L	L'opérande indique le bit dont l'état de signal est interrogé.



Flux d'énergie si :
- L'état de signal est 1 aux entrées E 0.0 ET E 0.1
- OU l'état de signal est 0 à l'entrée E 0.2

Description des bits du mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	–	–	–	–	–	x	x	x	1

Figure 8-2 Contact à ouverture

8.4 Sortie

Description

L'opération « Sortie » fonctionne comme une bobine dans un schéma à relais. La bobine au bout du circuit est excitée ou non selon les critères suivants :

- Si l'énergie peut traverser le circuit et atteindre la bobine (l'état de signal du circuit est 1), elle excite la bobine.
- Si l'énergie ne peut pas traverser tout le circuit et atteindre la bobine (l'état de signal du circuit est 0), elle ne peut pas exciter la bobine.

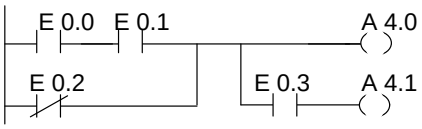
La séquence CONT représente le circuit. L'opération « Sortie » affecte l'état de signal de cette séquence à la bobine indiquée en opérande (ce qui revient à affecter l'état de signal du RLG à l'opérande). Si l'énergie traverse la séquence combinatoire, l'état de signal de la séquence est égal à 1 ; sinon, il est égal à 0. L'opération « Sortie » est affectée par le relais de masquage (MCR). Pour plus d'informations sur le fonctionnement du MCR, voir le paragraphe 20.5.

Vous ne pouvez placer une sortie qu'à l'extrémité droite d'une séquence combinatoire. Il est possible d'avoir plusieurs sorties. Vous ne pouvez pas placer une sortie seule dans un réseau autrement vide. La bobine doit avoir une liaison précédente.

Vous pouvez créer une sortie inversée à l'aide de l'opération « Inverser RLG ».

Tableau 8-3 Sortie : représentation et paramètre

Elément CONT	Paramètre	Type de données	Zone de mémoire	Description
<opérande> ——( )	<opérande>	BOOL	E, A, M, D, L	L'opérande indique le bit auquel est affecté l'état de signal de la séquence combinatoire.



L'état de signal de la sortie A 4.0 est 1 si :
- L'état de signal est 1 à l'entrée E 0.0 ET E 0.1
- OU l'état de signal est 0 à l'entrée E 0.2.

L'état de signal de la sortie A 4.1 est 1 si :
- L'état de signal est 1 à l'entrée E 0.0 ET E 0.1 et E 0.3
- OU l'état de signal est 0 à l'entrée E 0.2 et 1 à l'entrée E 0.3.

Description des bits du mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	–	–	–	–	–	0	x	–	0

Figure 8-3 Sortie

8.5 Connecteur

Description

L'opération « Connecteur » est un élément d'affectation intermédiaire qui mémorise le RLG. Cet élément sauvegarde la combinaison sur bits de la dernière branche ouverte jusqu'à ce que l'élément d'affectation soit atteint. En série avec d'autres contacts, « Connecteur » fonctionne comme un contact normal.

L'opération « Connecteur » est affectée par le relais de masquage (MCR). Pour plus d'informations sur le fonctionnement du MCR, voir le paragraphe 20.5.

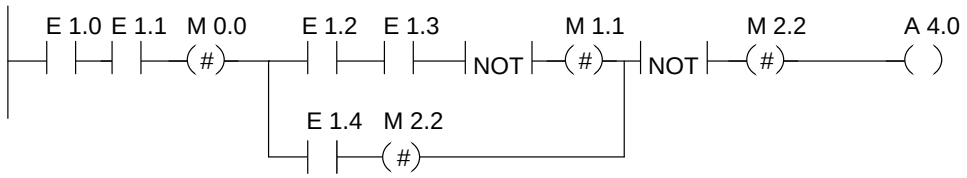
Lorsque vous placez un connecteur, vous devez tenir compte de certaines restrictions. Par exemple, ne placez jamais l'élément « Connecteur » à la fin d'une branche ouverte (voir également le paragraphe 6.1).

Vous pouvez créer une sortie inversée à l'aide de l'opération « Inverser RLG ».

Tableau 8-4 Connecteur : représentation et paramètre

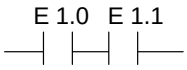
Elément CONT	Paramètre	Type de données	Zone de mémoire	Description
<opérande> —(#)	<opérande>	BOOL	E, A, M, D, L[1]	L'opérande indique le bit auquel est affecté le RLG.

[1] Dans l'opération « Connecteur », vous ne pouvez utiliser un opérande dans la zone de mémoire L que si vous le déclarez dans VAR_TEMP. Dans cette opération, vous ne pouvez pas utiliser la zone de mémoire L pour une adresse absolue.

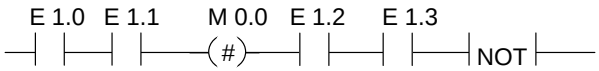


Les éléments « Connecteur » suivants ont le RLG ci-après :

M 0.0 a le RLG de



M 1.1 a le RLG de



M 2.2 a le RLG de la combinaison sur bits complète.

Description des bits du mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	–	–	–	–	–	0	x	–	0

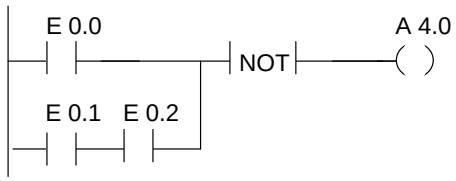
Figure 8-4 Connecteur

8.6 Inverser RLG

Description L'opération « Inverser RLG » inverse le RLG.

Tableau 8-5 Inverser RLG : représentation et paramètre

Elément CONT	Paramètre	Type de données	Zone de mémoire	Description
—| NOT |—	Néant	–	–	–



La sortie A 4.0 est à 1 si :

- L'état de signal est différent de 1 à l'entrée E 0.0
- OU l'état de signal est différent de 1 à l'entrée E 0.1 ET E 0.2.

Description des bits du mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	–	–	–	–	–	–	1	x	–

Figure 8-5 Inverser RLG

8.7 Sauvegarder RLG dans RB

Description

L’opération « Sauvegarder RLG dans RB » sauvegarde le RLG dans le bit RB du mot d’état.

Tableau 8-6 Sauvegarder RLG dans RB : représentation

Elément CONT	Paramètre	Type de données	Zone de mémoire	Description
SAVE	Néant	–	–	–

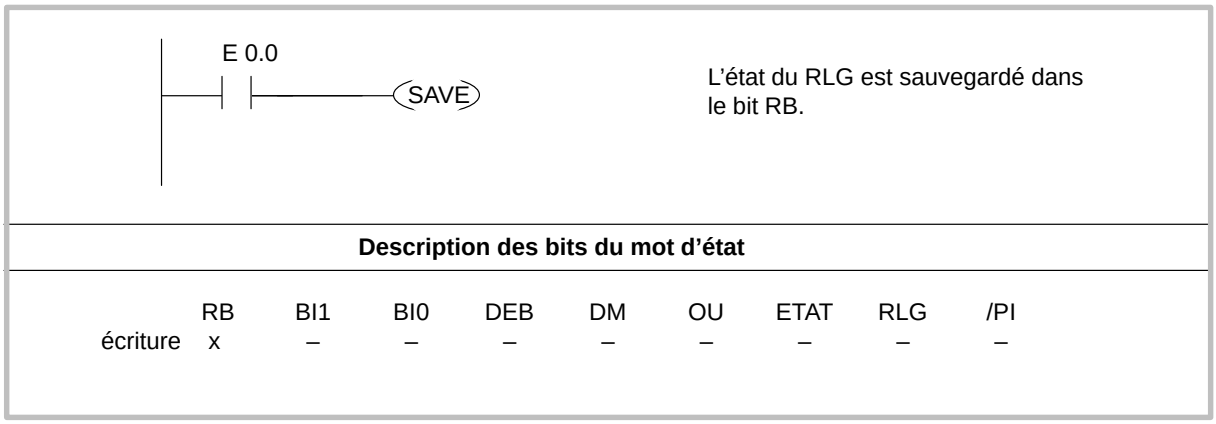

E 0.0
SAVE
L’état du RLG est sauvegardé dans le bit RB.
Description des bits du mot d’état
RB BI1 BI0 DEB DM OU ETAT RLG /PI
écriture x – – – – – – – –

Figure 8-6 Sauvegarder RLG dans RB

8.8 Mettre à 1

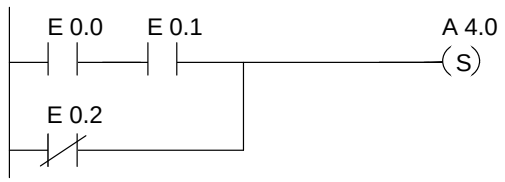
Description

L'opération « Mettre à 1 » ne s'exécute que si le RLG = 1. Dans ce cas, l'opération met son opérande à 1. Si le RLG = 0, l'opération n'a pas d'effet sur l'opérande précisé qui reste inchangé.

L'opération « Mettre à 1 » est affectée par le relais de masquage (MCR). Pour plus d'informations sur le fonctionnement du MCR, voir le paragraphe 20.5.

Tableau 8-7 Mettre à 1 : représentation et paramètre

Elément CONT	Paramètre	Type de données	Zone de mémoire	Description
<opérande> —(S)	<opérande>	BOOL	E, A, M, D, L	L'opérande indique le bit qui doit être mis à 1.



L'état de signal de la sortie A 4.0 est mis à 1 si :
- L'état de signal est 1 à l'entrée E 0.0 ET E 0.1
- OU l'état de signal est 0 à l'entrée E 0.2.

Si le RLG de la branche est égal à 0, l'état de signal de la sortie A 4.0 reste inchangé.

Description des bits du mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	–	–	–	–	–	0	x	–	0

Figure 8-7 Mettre à 1

8.9 Mettre à 0

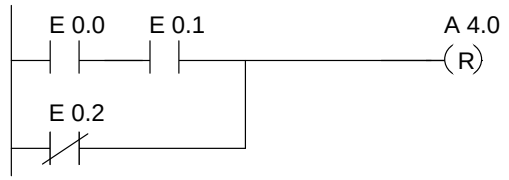
Description

L'opération « Mettre à 0 » ne s'exécute que si le RLG = 1. Dans ce cas, l'opération met son opérande à 0. Si le RLG = 0, l'opération n'a pas d'effet sur l'opérande précisé qui reste inchangé.

L'opération « Mettre à 0 » est affectée par le relais de masquage (MCR). Pour plus d'informations sur le fonctionnement du MCR, voir le paragraphe 20.5.

Tableau 8-8 Mettre à 0 : représentation et paramètre

Elément CONT	Paramètre	Type de données	Zone de mémoire	Description
<opérande> —(R)	<opérande>	BOOL TIMER COUNTER	E, A, M, T, Z, D, L	L'opérande indique le bit qui doit être mis à 0.



L'état de signal de la sortie A 4.0 est mis à 0 si :
- L'état de signal est 1 à l'entrée E 0.0 ET E 0.1
- OU l'état de signal est 0 à l'entrée E 0.2..

Si le RLG de la branche est égal à 0, l'état de signal de la sortie A 4.0 reste inchangé.

Description des bits du mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	–	–	–	–	–	0	x	–	0

Figure 8-8 Mettre à 0

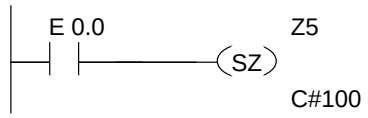
8.10 Initaliser compteur

Description

Vous pouvez utiliser l'opération « Initialiser compteur » pour affecter une valeur initiale au compteur que vous avez défini. L'opération « Initialiser compteur » ne s'exécute que si le RLG présente un front montant (c'est-à-dire s'il passe de 0 à 1).

Tableau 8-9 Initialiser compteur : représentation et paramètres, avec les abréviations SIMATIC et internationales

Elément CONT	Paramètre	Type de données	Zone de mémoire	Description
<opérande> —(SZ) —(SC) <valeur initiale>	Numéro de compteur	COUNTER	Z	L'opérande indique le numéro du compteur qui doit être initialisé.
	Valeur initiale	WORD	E, A. M, D, L	La valeur d'initialisation peut être comprise entre 0 et 999. La valeur doit être précédée par C# pour indiquer le format DCB, par exemple C#100.



Si l'état de signal de l'entrée E 0.0 passe de 0 à 1 (front montant du RLG), le compteur Z5 est initialisé avec la valeur 100. C# indique que vous entrez une valeur en format DCB.

En l'absence de front montant, la valeur de Z5 reste inchangée.

Description des bits du mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	–	–	–	–	–	0	–	–	0

Figure 8-9 Initialiser compteur

8.11 Incrémenter

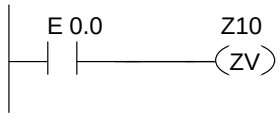
Description

L'opération « Incrémenter » incrémente d'un la valeur du compteur précisé si le RLG présente un front montant (c'est-à-dire s'il passe de 0 à 1) et si la valeur du compteur est inférieure à 999. En l'absence de front montant au RLG ou si le compteur est déjà égal à 999, la valeur du compteur reste inchangée.

L'opération « Initialiser compteur » initialise le compteur (voir paragraphe 8.10).

Tableau 8-10 Incrémenter : représentation et paramètre, avec les abréviations SIMATIC et internationales

Elément CONT	Paramètre	Type de données	Zone de mémoire	Description
<opérande> ZV CU	Numéro de compteur	COUNTER	Z	L'opérande indique le numéro du compteur à incrémenter.



Si l'état de signal de l'entrée E 0.0 passe de 0 à 1 (front montant du RLG), la valeur du compteur Z10 est incrémentée de 1, à moins qu'elle ne soit déjà égale à 999.

En l'absence de front montant, la valeur de Z10 reste inchangée.

Description des bits du mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	–	–	–	–	–	0	–	–	0

Figure 8-10 Incrémenter

8.12 Décrémenter

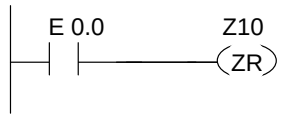
Description

L'opération « Décrémenter » décrémente d'un la valeur du compteur précisé si le RLG présente un front montant (c'est-à-dire s'il passe de 0 à 1) et si la valeur du compteur est supérieure à 0. En l'absence de front montant au RLG ou si le compteur est déjà égal à 0, la valeur du compteur reste inchangée.

L'opération « Initialiser compteur » initialise le compteur (voir paragraphe 8.10).

Tableau 8-11 Décrémenter : représentation et paramètre, avec les abréviations SIMATIC et internationales

Elément CONT	Paramètre	Type de données	Zone de mémoire	Description
<opérande> ——(ZR) ——(CD)	Numéro de compteur	COUNTER	Z	L'opérande indique le numéro du compteur à décrémenter.



Si l'état de signal de l'entrée E 0.0 passe de 0 à 1 (front montant du RLG), la valeur du compteur Z10 est décrémentée de 1, à moins qu'elle ne soit déjà égale à 0.

En l'absence de front montant, la valeur de Z10 reste inchangée.

Description des bits du mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	–	–	–	–	–	0	–	–	0

Figure 8-11 Décrémenter

8.13 Temporisation sous forme d'impulsion

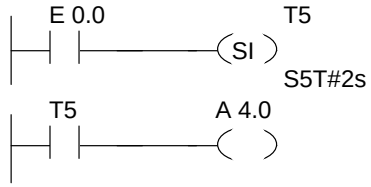
Description

L'opération « Temporisation sous forme d'impulsion » démarre la temporisation indiquée avec une valeur de temps donnée si le RLG présente un front montant (c'est-à-dire si le RLG passe de 0 à 1). La temporisation continue à s'exécuter tant que le RLG est positif. L'interrogation à 1 de l'état du signal de la temporisation fournit un résultat égal à 1 tant que la temporisation s'exécute. Si le RLG passe de 1 à 0 avant que le temps indiqué ne soit écoulé, la temporisation s'arrête. Dans ce cas, l'interrogation à 1 de l'état de signal fournit un résultat égal à 0.

Les unités de temps sont d (jours), h (heures), m (minutes), s (secondes) et ms (millisecondes). Les informations sur l'adresse en mémoire d'une temporisation ainsi que sur ses composants figurent au paragraphe 9.1.

Tableau 8-12 Temporisation sous forme d'impulsion : représentation et paramètres, avec les abréviations SIMATIC et internationales

Elément CONT	Paramètre	Type de données	Zone de mémoire	Description
<opérande> SI SP <valeur de temps>	Numéro de temporisation	TIMER	T	L'opérande indique le numéro de la temporisation à démarrer.
	Valeur de temps	S5TIME	E, A, M, D, L	Valeur de temps (format S5TIME)



Si l'état de signal de l'entrée E 0.0 passe de 0 à 1 (front montant du RLG), la temporisation T5 est démarrée. La temporisation continue à s'exécuter avec la valeur de temps précisée de 2 secondes tant que l'état de signal de l'entrée E 0.0 est égal à 1. Si l'état de signal de l'entrée E 0.0 passe de 1 à 0 avant expiration du temps précisé, la temporisation s'arrête.

L'état de signal à la sortie A 4.0 est 1 tant que la temporisation s'exécute.

Exemples de valeurs de temps :
S5T#2s = 2 secondes
S5T#12m_18s = 12 minutes et 18 secondes

Description des bits du mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	–	–	–	–	–	0	–	–	0

Figure 8-12 Temporisation sous forme d'impulsion

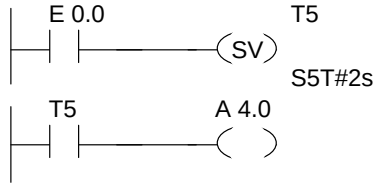
8.14 Temporisation sous forme d'impulsion prolongée

Description

L'opération « Temporisation sous forme d'impulsion prolongée » démarre la temporisation indiquée avec une valeur de temps donnée si le RLG présente un front montant (c'est-à-dire si le RLG passe de 0 à 1). La temporisation continue à s'exécuter même si le RLG passe à 0 avant que le temps précisé n'ait expiré. L'interrogation à 1 de l'état du signal de la temporisation fournit un résultat égal à 1 tant que la temporisation s'exécute. La temporisation est redémarrée (redéclenchée) avec le temps indiqué si le RLG passe de 0 à 1 alors que la temporisation s'exécute. Les informations sur l'adresse en mémoire d'une temporisation ainsi que sur ses composants figurent au paragraphe 9.1.

Tableau 8-13 Temporisation sous forme d'impulsion prolongée : représentation et paramètres, avec les abréviations SIMATIC et internationales

Elément CONT	Paramètre	Type de données	Zone de mémoire	Description
<opérande> ——(SV)	Numéro de temporisation	TIMER	T	L'opérande indique le numéro de la temporisation à démarrer.
——(SE) valeur de temps	Valeur de temps	S5TIME	E, A, M, D, L	Valeur de temps (format S5TIME)



Si l'état de signal de l'entrée E 0.0 passe de 0 à 1 (front montant du RLG), la temporisation T5 est démarrée. La temporisation continue à s'exécuter même en présence d'un front descendant du RLG. Si l'état de signal de l'entrée E 0.0 passe de 0 à 1 avant expiration du temps précisé, la temporisation est redéclenchée.

L'état de signal à la sortie A 4.0 est 1 tant que la temporisation s'exécute.

Description des bits du mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	–	–	–	–	–	0	–	–	0

Figure 8-13 Temporisation sous forme d'impulsion prolongée

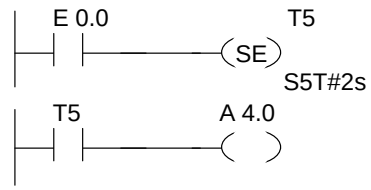
8.15 Temporisation sous forme de retard à la montée

Description

L'opération « Temporisation sous forme de retard à la montée » démarre la temporisation indiquée si le RLG présente un front montant (c'est-à-dire si le RLG passe de 0 à 1). L'interrogation à 1 de l'état du signal de la temporisation fournit un résultat égal à 1 lorsque le temps indiqué s'est écoulé sans erreur et que le RLG est toujours égal à 1. Si le RLG passe de 1 à 0 alors que la temporisation s'exécute, la temporisation est arrêtée. Dans ce cas, l'interrogation à 1 de l'état de signal fournit toujours un résultat égal à 0. Les informations sur l'adresse en mémoire d'une temporisation ainsi que sur ses composants figurent au paragraphe 9.1.

Tableau 8-14 Temporisation sous forme de retard à la montée : représentation et paramètres, avec les abréviations SIMATIC et internationales

Elément CONT	Paramètre	Type de données	Zone de mémoire	Description
<opérande> SE SD valeur de temps	Numéro de temporisation	TIMER	T	L'opérande indique le numéro de la temporisation à démarrer.
	Valeur de temps	S5TIME	E, A, M, D, L	Valeur de temps (format S5TIME)



Si l'état de signal de l'entrée E 0.0 passe de 0 à 1 (front montant du RLG), la temporisation T5 est démarrée. Si le temps expire et que l'état de signal de l'entrée E 0.0 est toujours 1, la sortie A 4.0 est mise à 1. Si l'état de signal de l'entrée E 0.0 passe de 1 à 0, la temporisation est arrêtée et la sortie A 4.0 est à 0.

Description des bits du mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	–	–	–	–	–	0	–	–	0

Figure 8-14 Temporisation sous forme de retard à la montée

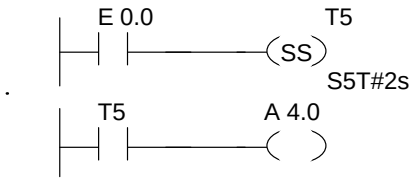
8.16 Temporisation sous forme de retard à la montée mémorisé

Description

L'opération « Temporisation sous forme de retard à la montée mémorisé » démarre la temporisation indiquée si le RLG présente un front montant (c'est-à-dire si le RLG passe de 0 à 1). La temporisation continue à s'exécuter même si le RLG passe à 0 avant que le temps n'expire. L'interrogation à 1 de l'état du signal de la temporisation fournit un résultat égal à 1 lorsque le temps indiqué s'est écoulé, quel que soit le RLG. Si le RLG passe de 0 à 1 alors que la temporisation s'exécute, la temporisation est redémarrée (redéclenchée) avec le temps indiqué. Les informations sur l'adresse en mémoire d'une temporisation ainsi que sur ses composants figurent au paragraphe 9.1.

Tableau 8-15 Temporisation sous forme de retard à la montée mémorisé : représentation et paramètres

Elément CONT	Paramètre	Type de données	Zone de mémoire	Description
<opérande> ——(SS) valeur de temps	Numéro de temporisation	TIMER	T	L'opérande indique le numéro de la temporisation à démarrer.
	Valeur de temps	S5TIME	E, A, M, D, L	Valeur de temps (format S5TIME)



Si l'état de signal de l'entrée E 0.0 passe de 0 à 1 (front montant du RLG), la temporisation T5 est démarrée. La temporisation continue à s'exécuter même si l'état de signal passe de 1 à 0 à l'entrée E 0.0. Si l'état de signal de l'entrée E 0.0 passe de 0 à 1 avant que le temps n'ait expiré, la temporisation est redéclenchée. L'état de signal de la sortie A 4.0 est à 1 si le temps a expiré.

Description des bits du mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	–	–	–	–	–	0	–	–	0

Figure 8-15 Temporisation sous forme de retard à la montée mémorisé

8.17 Temporisation sous forme de retard à la retombée

Description

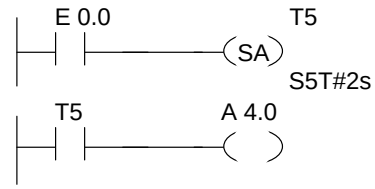
L'opération « Temporisation sous forme de retard à la retombée » démarre la temporisation indiquée si le RLG présente un front descendant (c'est-à-dire si le RLG passe de 1 à 0). L'interrogation à 1 de l'état du signal de la temporisation fournit un résultat égal à 1 lorsque le RLG est égal à 1 ou que la temporisation s'exécute. La temporisation est mise à 0 lorsque le RLG passe de 0 à 1 alors que la temporisation s'exécute. La temporisation n'est redémarrée que lorsque le RLG repasse de 1 à 0.

Paramètre

Les informations sur l'adresse en mémoire d'une temporisation ainsi que sur ses composants figurent au paragraphe 9.1.

Tableau 8-16 Temporisation sous forme de retard à la retombée : représentation et paramètres, avec les abréviations SIMATIC et internationales

Elément CONT	Paramètre	Type de données	Zone de mémoire	Description
<opérande> ——(SA) ——(SF) valeur de temps	Numéro de temporisation	TIMER	T	L'opérande indique le numéro de la temporisation à démarrer.
	Valeur de temps	S5TIME	E, A, M, D, L	Valeur de temps (format S5TIME)



Si l'état de signal de l'entrée E 0.0 passe de 1 à 0, la temporisation T5 est démarrée.

Si l'état de signal de l'entrée E 0.0 passe de 0 à 1, la temporisation est mise à 0.

L'état de signal de la sortie A 4.0 est à 1 lorsque l'état de signal de l'entrée E 0.0 est 1 ou que la temporisation s'exécute.

Description des bits du mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	–	–	–	–	–	0	–	–	0

Figure 8-16 Temporisation sous forme de retard à la retombée

8.18 Détecter front montant du RLG

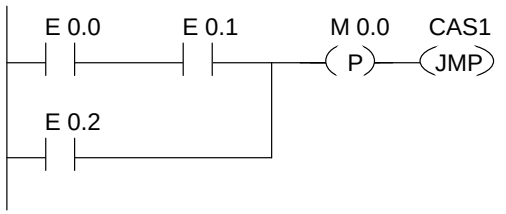
Description

L'opération « Détecter front montant du RLG » détecte le passage de 0 à 1 de l'état de signal du RLG et montre cette transition avec un RLG égal à 1 après cette opération. L'état du signal actuel du RLG est comparé à celui de l'opérande, en l'occurrence le mémento de front. Si l'état du signal de l'opérande est égal à 0 et si le RLG est égal à 1 avant l'opération « Détecter front montant du RLG », le RLG passe à 1 (impulsion) après l'opération en question. Dans tout autre cas, le RLG est égal à 0. Le RLG avant l'opération est sauvegardé dans l'opérande.

Lorsque vous placez l'élément « Détecter front montant du RLG », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 8-17 Détecter front montant du RLG : représentation et paramètre

Elément CONT	Paramètre	Type de données	Zone de mémoire	Description
<opérande1> —(P)	<opérande1>	BOOL	A, M, D, L	L'opérande indique le mémento de front qui mémorise l'ancien RLG.



Le mémento de front M 0.0 mémorise l'ancien état de signal du RLG provenant de la combinaison sur bits entière. En cas de passage de 0 à 1 du RLG, le programme effectue un saut au repère CAS1.

Description des bits du mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	–	–	–	–	–	0	x	x	1

Figure 8-17 Détecter front montant du RLG

8.19 Détecter front descendant du RLG

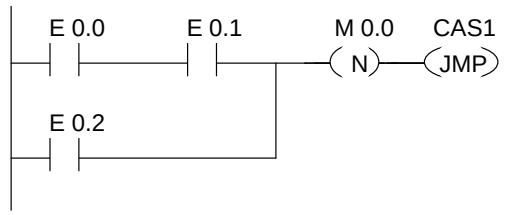
Description

L'opération « Détecter front descendant du RLG » détecte le passage de 1 à 0 de l'état de signal du RLG et montre cette transition avec un RLG égal à 1 après cette opération. L'état du signal actuel du RLG est comparé à celui de l'opérande, en l'occurrence le mémento de front. Si l'état du signal de l'opérande est égal à 1 et si le RLG est égal à 0 avant l'opération « Détecter front montant du RLG », le RLG passe à 0 (impulsion) après l'opération en question. Dans tout autre cas, le RLG est égal à 1. Le RLG avant l'opération est sauvegardé dans l'opérande.

Lorsque vous placez l'élément « Détecter front descendant du RLG », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 8-18 Détecter front descendant du RLG : représentation et paramètre

Elément CONT	Paramètre	Type de données	Zone de mémoire	Description
<opérande1> —(N)	<opérande1>	BOOL	A, M, D	L'opérande indique le mémento de front qui mémorise l'ancien RLG.



Le mémento de front M 0.0 mémorise l'ancien état de signal du RLG provenant de la combinaison sur bits entière. En cas de passage de 1 à 0 du RLG, le programme effectue un saut au repère CAS1.

Description des bits du mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	–	–	–	–	–	0	x	x	1

Figure 8-18 Détecter front descendant du RLG

8.20 Détecter front montant de signal

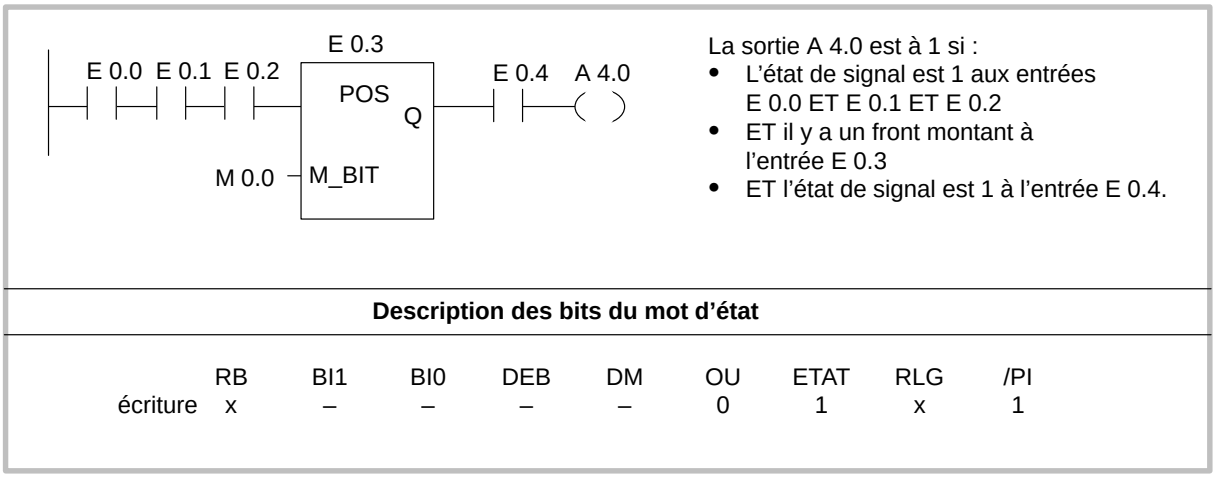
Description

L'opération « Détecter front montant de signal » compare l'état de signal de <opérande1> à celui provenant de l'interrogation d'état de signal précédente figurant dans <opérande2>. En cas de passage de 0 à 1, la sortie Q est mise à 1. Dans tout autre cas, elle est mise à 0.

Lorsque vous placez le pavé « Détecter front montant de signal », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 8-19 Détecter front montant de signal : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
<opérande1> POS Q <opérande2> M_BIT	<opérande1>	BOOL	E, A, M, D, L	Signal à interroger pour détecter un front montant
	M_BIT	BOOL	A, M, D	L'opérande M_BIT indique le mémento de front qui mémorise l'état de signal précédent de POS. N'utilisez la mémoire image des entrées (E) pour M_BIT que si cet opérande n'est pas déjà occupé par un module d'entrées.
	Q	BOOL	E, A, M, D, L	Sortie monostable


E 0.3
E 0.0 E 0.1 E 0.2
POS
Q
E 0.4 A 4.0
M 0.0
M_BIT
La sortie A 4.0 est à 1 si :
• L'état de signal est 1 aux entrées E 0.0 ET E 0.1 ET E 0.2
• ET il y a un front montant à l'entrée E 0.3
• ET l'état de signal est 1 à l'entrée E 0.4.

Description des bits du mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	–	–	–	–	0	1	x	1

Figure 8-19 Détecter front montant de signal

8.21 Détecter front descendant de signal

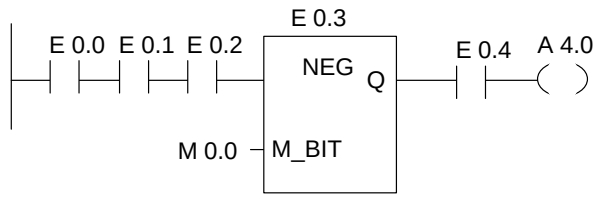
Description

L'opération « Détecter front descendant de signal » compare l'état de signal de <opérande1> à celui provenant de l'interrogation d'état de signal précédente figurant dans <opérande2>. En cas de passage de 1 à 0, la sortie Q est mise à 1. Dans tout autre cas, elle est mise à 0.

Lorsque vous placez le pavé « Détecter front descendant de signal », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 8-20 Détecter front descendant de signal : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
<opérande1> NEG Q <opérande2> M_BIT	<opérande1>	BOOL	E, A, M, D, L	Signal à interroger pour détecter un front descendant
	M_BIT	BOOL	A, M, D	L'opérande M_BIT indique le mémento de front qui mémorise l'état de signal précédent de NEG. N'utilisez la mémoire image des entrées (E) pour M_BIT que si cet opérande n'est pas déjà occupé par un module d'entrées.
	Q	BOOL	E, A, M, D, L	Sortie monostable



La sortie A 4.0 est à 1 si :
- L'état de signal est 1 aux entrées E 0.0 ET E 0.1 ET E 0.2
- ET il y a un front descendant à l'entrée E 0.3
- ET l'état de signal est 1 à l'entrée E 0.4.

Description des bits du mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	–	–	–	–	0	1	x	1

Figure 8-20 Détecter front descendant de signal

8.22 Bascule mise à 1, mise à 0

Description

L'opération « Bascule mise à 1, mise à 0 » n'exécute la mise à 1 (S) et la mise à 0 (R) que lorsque le RLG est égal à 1. Lorsque le RLG est égal à 0, l'opérande précisé dans l'opération reste inchangé.

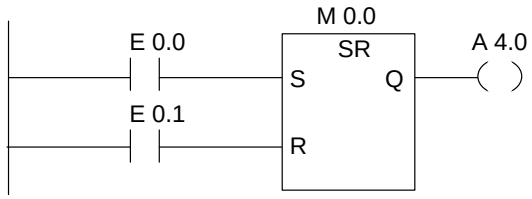
Une « Bascule mise à 1, mise à 0 » est mise à 1 si l'état de signal est 1 à l'entrée S et 0 à l'entrée R. Si l'état de signal est 0 à l'entrée S et 1 à l'entrée R, la bascule est mise à 0. Si le RLG est égal à 1 aux deux entrées, la bascule est mise à 0.

L'opération « Bascule mise à 1, mise à 0 » est affectée par le relais de masquage (MCR). Pour plus d'informations sur le fonctionnement du MCR, voir le paragraphe 20.5.

Lorsque vous placez le pavé « Bascule mise à 1, mise à 0 », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 8-21 Bascule mise à 1, mise à 0 : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
<opérande> SR S Q R	<opérande>	BOOL	E, A, M, D, L	L'opérande indique le bit qui doit être mis à 1 ou à 0.
	S	BOOL	E, A, M, D, L	Mise à 1 activée
	R	BOOL	E, A, M, D, L	Mise à 0 activée
	Q	BOOL	E, A, M, D, L	Etat de signal de <opérande>



Si l'état de signal est 1 à l'entrée E 0.0 et 0 à l'entrée E 0.1, le bit de mémento M 0.0 est mis à 1 et la sortie A 4.0 est à 1.

Si l'état de signal est 0 à l'entrée E 0.0 et 1 à l'entrée E 0.1, le bit de mémento M 0.0 est mis à 0 et la sortie A 4.0 est à 0.

Si les deux états de signal ont la valeur 0, rien ne se passe. En revanche, s'ils ont tous deux la valeur 1, la mise à 0, exécutée en dernier, l'emporte : M 0.0 est mis à 0 et la sortie A 4.0 est à 0.

Description des bits du mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	–	–	–	–	–	x	x	x	1

Figure 8-21 Bascule mise à 1, mise à 0

8.23 Bascule mise à 0, mise à 1

Description

L'opération « Bascule mise à 0, mise à 1 » n'exécute la mise à 1 (S) et la mise à 0 (R) que lorsque le RLG est égal à 1. Lorsque le RLG est égal à 0, l'opérande précisé dans l'opération reste inchangé.

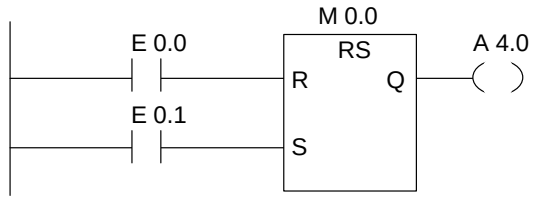
Une « Bascule mise à 0, mise à 1 » est mise à 1 si l'état de signal est 1 à l'entrée R et 0 à l'entrée S. Si l'état de signal est 0 à l'entrée R et 1 à l'entrée S, la bascule est mise à 1. Si le RLG est égal à 1 aux deux entrées, la bascule est mise à 1.

L'opération « Bascule mise à 0, mise à 1 » est affectée par le relais de masquage (MCR). Pour plus d'informations sur le fonctionnement du MCR, voir le paragraphe 20.5.

Lorsque vous placez le pavé « Bascule mise à 0, mise à 1 », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 8-22 Bascule mise à 0, mise à 1 : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
<opérande> RS R Q S	<opérande>	BOOL	E, A, M, D, L	L'opérande indique le bit qui doit être mis à 1 ou à 0.
	S	BOOL	E, A, M, D, L	Mise à 0 activée
	R	BOOL	E, A, M, D, L	Mise à 1 activée
	Q	BOOL	E, A, M, D, L	Etat de signal de <opérande>



Si l'état de signal est 1 à l'entrée E 0.0 et 0 à l'entrée E 0.1, le bit de mémento M 0.0 est mis à 0 et la sortie A 4.0 est à 0.

Si l'état de signal est 0 à l'entrée E 0.0 et 1 à l'entrée E 0.1, le bit de mémento M 0.0 est mis à 1 et la sortie A 4.0 est à 1.

Si les deux états de signal ont la valeur 0, rien ne se passe. En revanche, s'ils ont tous deux la valeur 1, la mise à 1, exécutée en dernier, l'emporte : M 0.0 est mis à 1 et la sortie A 4.0 est à 1.

Description des bits du mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	–	–	–	–	–	x	x	x	1

Figure 8-22 Bascule mise à 0, mise à 1

Opérations de temporisation

9

Contenu de ce chapitre

Paragraphe	Thème	Page
9.1	Adresse d’une temporisation en mémoire et composants d’une temporisation	9-2
9.2	Choix de la temporisation correcte	9-4
9.3	Temporisation sous forme d’impulsion	9-5
9.4	Temporisation sous forme d’impulsion prolongée	9-7
9.5	Temporisation sous forme de retard à la montée	9-9
9.6	Temporisation sous forme de retard à la montée mémorisé	9-11
9.7	Temporisation sous forme de retard à la retombée	9-13

9.1 Adresse d'une temporisation en mémoire et composants d'une temporisation

Zone de mémoire

Une zone de mémoire est réservée aux temporisations dans votre CPU. Un mot de 16 bits y est réservé pour chaque opérande de temporisation. La programmation en CONT permet d'utiliser jusqu'à 256 temporisations. Le nombre de mots de temporisation disponibles dans votre CPU figure dans les caractéristiques de la CPU.

Les fonctions suivantes ont accès à la zone de mémoire réservée aux temporisations :

- opérations de temporisation,
- actualisation des mots de temporisation avec une horloge. Cette fonction décrémente, à l'état de marche (RUN) de la CPU, une valeur donnée d'une unité dans un intervalle défini par la base de temps, et ce, jusqu'à ce que la valeur de temps soit égale à zéro.

Valeur de temps

La valeur de temps est contenue sous forme binaire dans les bits 0 à 9 du mot de temporisation. Elle détermine un nombre d'unités. L'actualisation de l'heure décrémente la valeur de temps d'une unité dans un intervalle défini par la base de temps. La décrémentation se poursuit jusqu'à ce que la valeur de temps soit égale à zéro. Pour charger une valeur de temps (voir figure 9-1), vous pouvez utiliser le format binaire, hexadécimal ou décimal codé binaire (DCB). La plage de temps est comprise entre 0 et 9 990 secondes.

Vous pouvez charger une valeur de temps prédéfinie en utilisant l'un des deux formats suivants :

- W#16#wxyz où
  - w = base de temps (c'est-à-dire l'intervalle de temps ou la résolution)
  - xyz = valeur de temps en format décimal codé binaire (DCB)
- S5T#aH_bbM_ccS_dddMS où
  - a = heures, bb = minutes, cc = secondes et ddd = millisecondes
  - la base de temps est choisie automatiquement et la valeur est arrondie au nombre inférieur le plus proche avec cette base de temps.

La valeur de temps maximale que vous pouvez indiquer est égale à 9 990 secondes ou 2H_46M_30S.

Base de temps

La base de temps est contenue en code binaire dans les bits 12 et 13 du mot de temporisation. Elle détermine à quel intervalle la valeur de temps va être décrémentée (voir tableau 9-1 et figure 9-1). La base de temps minimale est égale à 10 ms ; la base de temps maximale à 10 s.

Tableau 9-1 Base de temps avec son code binaire

Base de temps	Code binaire de la base de temps
10 ms	00
100 ms	01
1 s	10
10 s	11

Puisque les valeurs de temps sont mémorisées dans un seul intervalle de temps, celles qui ne sont pas un multiple exact de cet intervalle sont tronquées. Les valeurs dont la résolution est trop grande pour la plage considérée sont arrondies de sorte à entrer dans la plage voulue, mais pas avec la résolution souhaitée. Le tableau 9-2 montre les résolutions possibles avec les plages correspondantes.

Tableau 9-2 Résolutions et plages de la base de temps

Résolution	Plage
0,01 seconde	10MS à 9S_990MS
0,1 seconde	100MS à 1M_39S_900MS
1 seconde	1S à 16M_39S
10 secondes	10S à 2HR_46M_30S

Configuration des bits dans la cellule de temporisation

Lorsqu'une temporisation est démarrée, le contenu de la cellule de temporisation est utilisé comme valeur de temps. Les bits 0 à 11 de la cellule de temporisation contiennent la valeur de temps en format décimal codé binaire (format DCB : chaque groupe de quatre bits contient le code binaire d'une valeur décimale). Les bits 12 et 13 contiennent la base de temps en code binaire (voir tableau 9-1). La figure 9-1 montre le contenu de la cellule de temporisation dans laquelle vous avez chargé la valeur de temps 127 et une base de temps de 1 seconde.

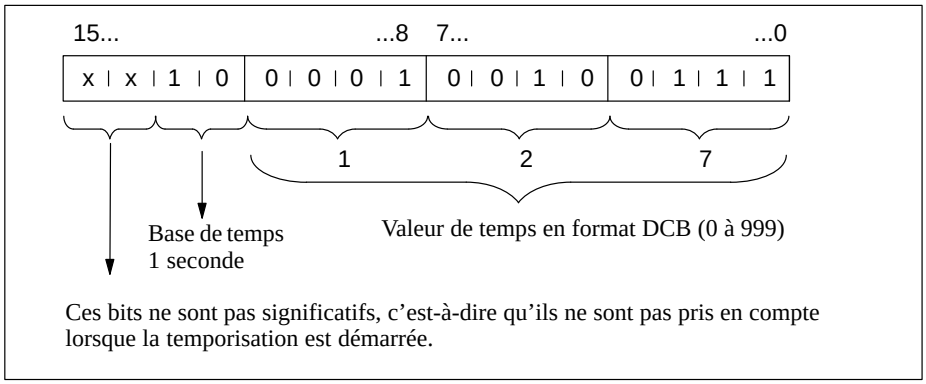


Figure 9-1 Contenu de la cellule de temporisation : valeur de temps 127, base de temps 1 s

Lecture de la valeur et de la base de temps

Chaque pavé de temporisation possède deux sorties, DUAL (BI) et DEZ (BCD), pour lesquelles vous pouvez indiquer une adresse de mot. La sortie DUAL fournit la valeur de temps en format binaire. La sortie DEZ fournit la base de temps et la valeur de temps en format décimal codé binaire (DCB).

9.2 Choix de la temporisation correcte

La figure 9-2 donne une vue d'ensemble des cinq types de temporisations décrites dans ce chapitre. Elle doit vous aider à choisir la temporisation qui répond le mieux à vos besoins.

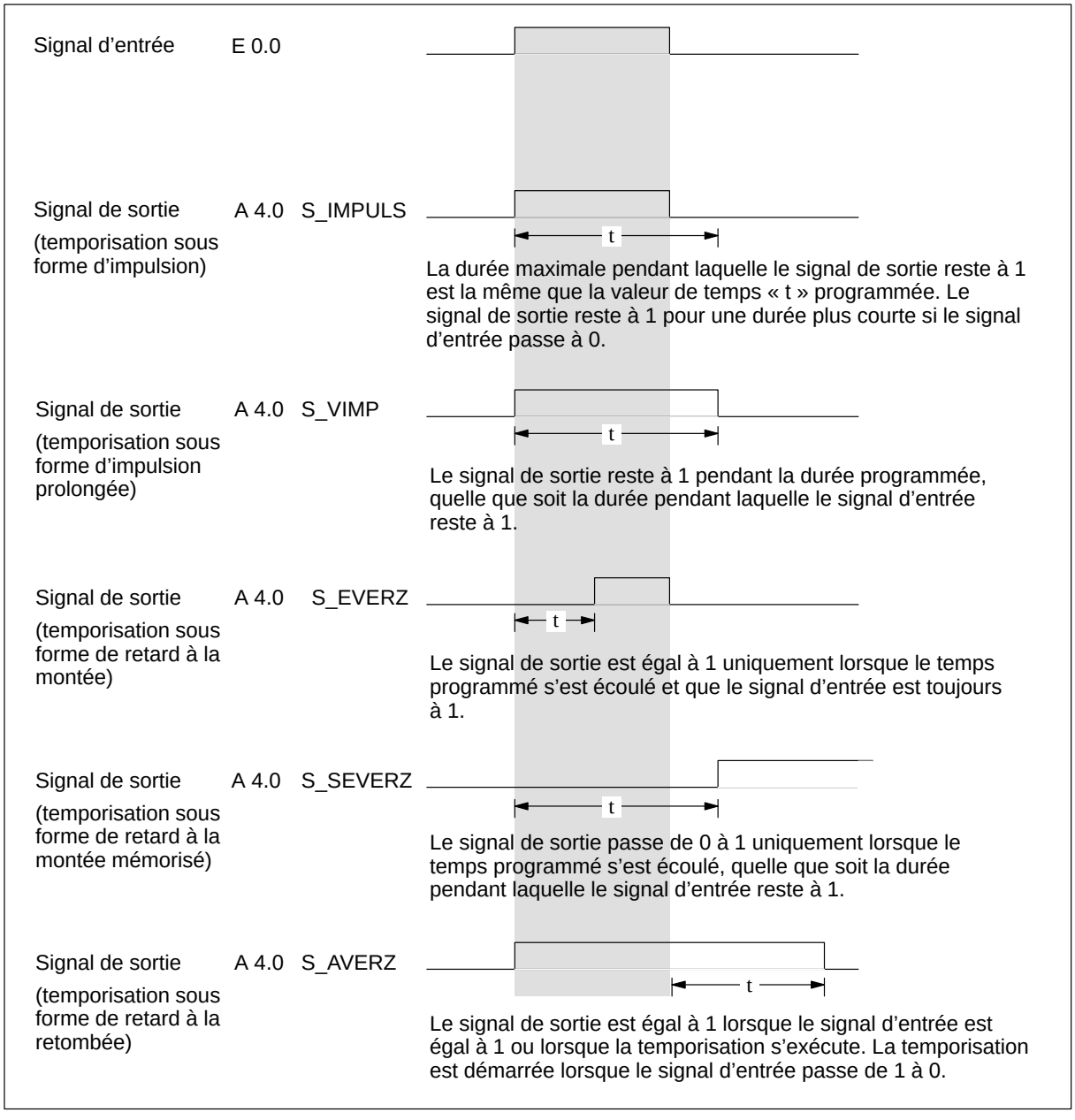

Signal d'entrée E 0.0
Signal de sortie A 4.0 S_IMPULS
(temporisation sous forme d'impulsion)
t
La durée maximale pendant laquelle le signal de sortie reste à 1 est la même que la valeur de temps « t » programmée. Le signal de sortie reste à 1 pour une durée plus courte si le signal d'entrée passe à 0.
Signal de sortie A 4.0 S_VIMP
(temporisation sous forme d'impulsion prolongée)
t
Le signal de sortie reste à 1 pendant la durée programmée, quelle que soit la durée pendant laquelle le signal d'entrée reste à 1.
Signal de sortie A 4.0 S_EVERZ
(temporisation sous forme de retard à la montée)
t
Le signal de sortie est égal à 1 uniquement lorsque le temps programmé s'est écoulé et que le signal d'entrée est toujours à 1.
Signal de sortie A 4.0 S_SEVERZ
(temporisation sous forme de retard à la montée mémorisé)
t
Le signal de sortie passe de 0 à 1 uniquement lorsque le temps programmé s'est écoulé, quelle que soit la durée pendant laquelle le signal d'entrée reste à 1.
Signal de sortie A 4.0 S_AVERZ
(temporisation sous forme de retard à la retombée)
t
Le signal de sortie est égal à 1 lorsque le signal d'entrée est égal à 1 ou lorsque la temporisation s'exécute. La temporisation est démarrée lorsque le signal d'entrée passe de 1 à 0.

Figure 9-2 Choix de la temporisation correcte

9.3 Temporisation sous forme d'impulsion

Description

L'opération « Temporisation sous forme d'impulsion » démarre la temporisation précisée en cas de front montant (c'est-à-dire lorsque l'état de signal passe de 0 à 1) à l'entrée de démarrage S. Un changement d'état de signal est toujours nécessaire pour démarrer une temporisation. La valeur de temps indiquée à l'entrée TW s'écoule pendant le temps programmé, tant que l'état de signal à l'entrée S est égal à 1. Pendant que la temporisation s'exécute, l'interrogation à 1 de l'état de signal à la sortie Q donne 1 comme résultat. En cas de passage de 1 à 0 à l'entrée S avant que le temps n'ait expiré, la temporisation s'arrête. Dans ce cas, l'interrogation à 1 de l'état de signal à la sortie Q donne 0 comme résultat.

En cas de passage de 0 à 1 à l'entrée de remise à zéro R pendant que la temporisation s'exécute, cette dernière est remise à zéro. Cette transition remet aussi le temps et la base de temps à zéro. L'état de signal 1 à l'entrée R de la temporisation n'a aucun effet si la temporisation ne s'exécute pas.

La valeur de temps en cours peut être lue en format binaire à la sortie DUAL et en format décimal codé binaire à la sortie DEZ.

Tableau 9-3 Temporisation sous forme d'impulsion : représentation et paramètres, avec les abréviations SIMATIC

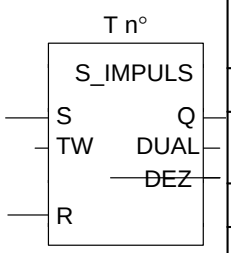
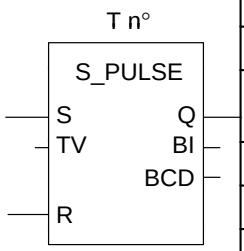
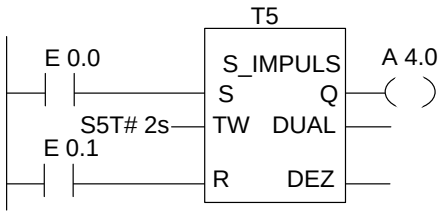
Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	n°	TIMER	T	Numéro d'identification de la temporisation. La plage dépend de la CPU.
	S	BOOL	E, A, M, D, L, T, Z	Entrée de démarrage
	TW	S5TIME	E, A, M, D, L	Valeur de temps prédéfinie (plage : 0 à 9999)
	R	BOOL	E, A, M, D, L, T, Z	Entrée de remise à zéro
	Q	BOOL	E, A, M, D, L	Etat de la temporisation
	DUAL	WORD	E, A, M, D, L	Valeur de temps restante (format binaire)
	DEZ	WORD	E, A, M, D, L	Valeur de temps restante (format DCB)

Tableau 9-4 Temporisation sous forme d'impulsion : représentation et paramètres, avec les abréviations internationales

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	n°	TIMER	T	Numéro d'identification de la temporisation. La plage dépend de la CPU.
	S	BOOL	E, A, M, D, L, T, Z	Entrée de démarrage
	TV	S5TIME	E, A, M, D, L	Valeur de temps prédéfinie (plage : 0 à 9999)
	R	BOOL	E, A, M, D, L, T, Z	Entrée de remise à zéro
	Q	BOOL	E, A, M, D, L	Etat de la temporisation
	BI	WORD	E, A, M, D, L	Valeur de temps restante (format binaire)
	BCD	WORD	E, A, M, D, L	Valeur de temps restante (format DCB)

Exemple

La figure 9-3 montre l'opération « Temporisation sous forme d'impulsion », décrit les bits du mot d'état et souligne les caractéristiques de ce type de temporisation. Lorsque vous placez un pavé de temporisation, vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).



La temporisation T 5 est démarrée si l'état de signal passe de 0 à 1 à l'entrée E 0.0 (front montant du RLG). Elle s'exécute avec la valeur indiquée, égale à 2 secondes (2s), tant que E 0.0 est à 1. Si l'état de signal de l'entrée E 0.0 passe de 1 à 0 avant que le temps n'ait expiré, la temporisation s'arrête. Si l'état de signal de l'entrée E 0.1 passe de 0 à 1 alors que la temporisation s'exécute, la temporisation est remise à zéro. L'état de signal à la sortie A 4.0 est 1 tant que la temporisation s'exécute.

Exemples pour d'autres valeurs de temps :
Unités disponibles : h (heures), m (minutes), s (secondes), ms (millisecondes)

S5T#4s —> 4 secondes
S5T#1h_15m —> 1 heure et 15 minutes
S5T#2h_46m_30s—> 2 heures, 46 minutes et 30 secondes

Description des bits du mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	–	–	–	–	–	x	x	x	1

Chronogramme

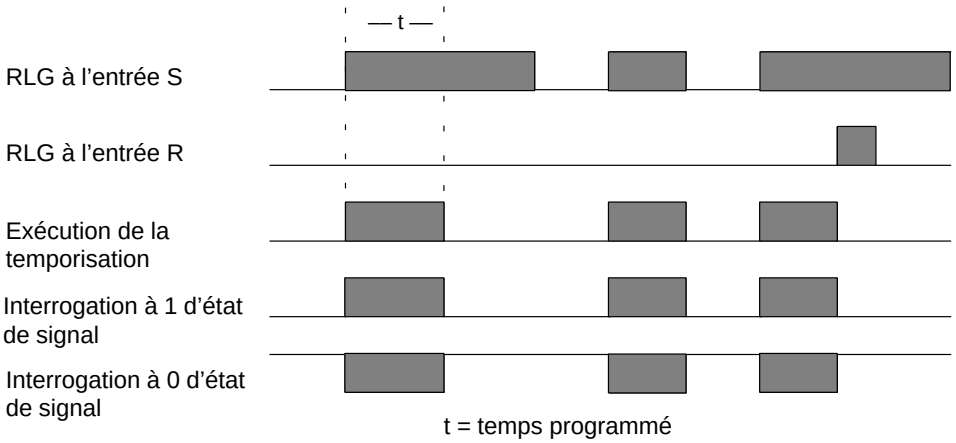

— t —
RLG à l'entrée S
RLG à l'entrée R
Exécution de la temporisation
Interrogation à 1 d'état de signal
Interrogation à 0 d'état de signal
t = temps programmé

Figure 9-3 Temporisation sous forme d'impulsion

9.4 Temporisation sous forme d'impulsion prolongée

Description

L'opération « Temporisation sous forme d'impulsion prolongée » démarre la temporisation précisée en cas de front montant (c'est-à-dire lorsque l'état de signal passe de 0 à 1) à l'entrée de démarrage S. Un changement d'état de signal est toujours nécessaire pour démarrer une temporisation. La valeur de temps indiquée à l'entrée TW continue à s'écouler même si l'état de signal à l'entrée S passe à 0 avant expiration du temps. Tant que la temporisation s'exécute, l'interrogation à 1 de l'état de signal à la sortie Q donne 1 comme résultat. La temporisation est redémarrée avec la valeur de temps indiquée si l'état de signal à l'entrée S passe de 0 à 1 alors que la temporisation s'exécute.

En cas de passage de 0 à 1 à l'entrée de remise à zéro (R) pendant que la temporisation s'exécute, cette dernière est remise à zéro. Cette transition remet aussi le temps et la base de temps à zéro.

La valeur de temps en cours peut être lue en format binaire à la sortie DUAL et en format décimal codé binaire à la sortie DEZ.

Tableau 9-5 Temporisation sous forme d'impulsion prolongée : représentation et paramètres, avec les abréviations SIMATIC

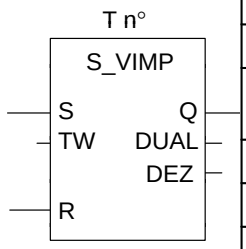
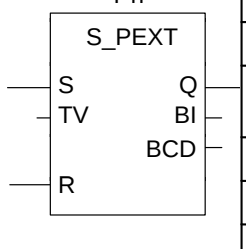
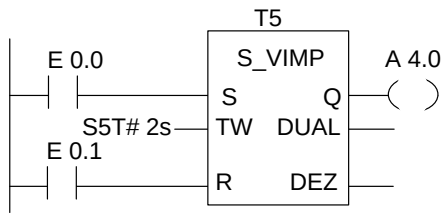
Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	n°	TIMER	T	Numéro d'identification de la temporisation. La plage dépend de la CPU.
	S	BOOL	E, A, M, D, L, T, Z	Entrée de démarrage
	TW	S5TIME	E, A, M, D, L	Valeur de temps prédéfinie (plage : 0 à 9999)
	R	BOOL	E, A, M, D, L, T, Z	Entrée de remise à zéro
	Q	BOOL	E, A, M, D, L	Etat de la temporisation
	DUAL	WORD	E, A, M, D, L	Valeur de temps restante (format binaire)
	DEZ	WORD	E, A, M, D, L	Valeur de temps restante (format DCB)

Tableau 9-6 Temporisation sous forme d'impulsion prolongée : représentation et paramètres, avec les abréviations internationales

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	n°	TIMER	T	Numéro d'identification de la temporisation. La plage dépend de la CPU.
	S	BOOL	E, A, M, D, L, T, Z	Entrée de démarrage
	TV	S5TIME	E, A, M, D, L	Valeur de temps prédéfinie (plage : 0 à 9999)
	R	BOOL	E, A, M, D, L, T, Z	Entrée de remise à zéro
	Q	BOOL	E, A, M, D, L	Etat de la temporisation
	BI	WORD	E, A, M, D, L	Valeur de temps restante (format binaire)
	BCD	WORD	E, A, M, D, L	Valeur de temps restante (format DCB)

Exemple

La figure 9-4 montre l'opération « Temporisation sous forme d'impulsion prolongée », décrit les bits du mot d'état et souligne les caractéristiques de ce type de temporisation. Lorsque vous placez un pavé de temporisation, vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

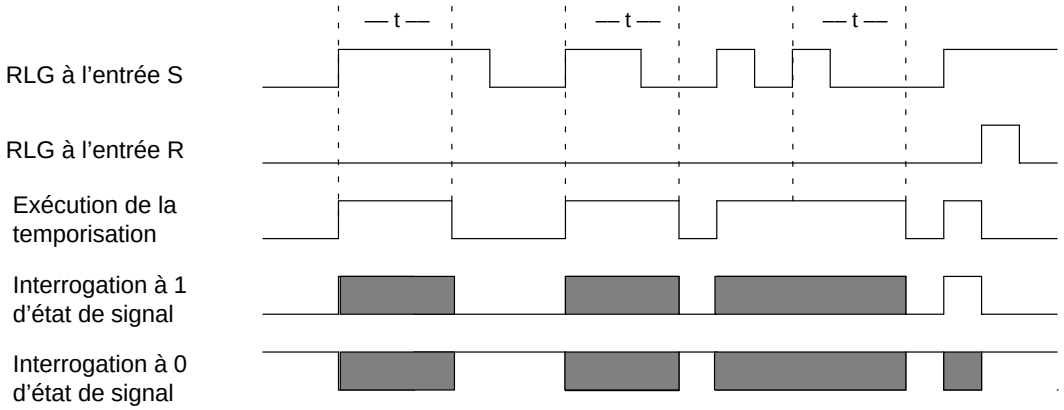


La temporisation T 5 est démarrée si l'état de signal passe de 0 à 1 à l'entrée E 0.0 (front montant du RLG). Le temps de deux secondes (2s) indiqué continue à s'écouler même en cas de front descendant à l'entrée S. Si l'état de signal de l'entrée E 0.0 passe de 0 à 1 avant que le temps n'ait expiré, la temporisation est redémarrée. Si l'état de signal de l'entrée E 0.1 passe de 0 à 1 alors que la temporisation s'exécute, la temporisation est redémarrée. L'état de signal à la sortie A 4.0 est 1 tant que la temporisation s'exécute (voir aussi paragraphe 9.3).

Description des bits du mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	–	–	–	–	–	x	x	x	1

Chronogramme


— t —
— t —
— t —
RLG à l'entrée S
RLG à l'entrée R
Exécution de la temporisation
Interrogation à 1 d'état de signal
Interrogation à 0 d'état de signal

t = temps programmé

Figure 9-4 Temporisation sous forme d'impulsion prolongée

9.5 Temporisation sous forme de retard à la montée

Description

L'opération « Temporisation sous forme de retard à la montée » démarre la temporisation précisée en cas de front montant (c'est-à-dire lorsque l'état de signal passe de 0 à 1) à l'entrée de démarrage S. Un changement d'état de signal est toujours nécessaire pour démarrer une temporisation. La valeur de temps indiquée à l'entrée TW s'écoule tant que l'état de signal à l'entrée S est à 1. L'interrogation à 1 de l'état de signal à la sortie Q donne 1 comme résultat lorsque le temps a expiré sans erreur, que l'état de signal à l'entrée S est toujours 1 et que l'entrée de remise à zéro (R) reste à 0. La temporisation s'arrête si l'état de signal à l'entrée S passe de 1 à 0 alors que la temporisation s'exécute. Dans ce cas, l'interrogation à 1 de l'état de signal à la sortie Q donne toujours 0 comme résultat.

En cas de passage de 0 à 1 à l'entrée de remise à zéro (R) pendant que la temporisation s'exécute, cette dernière est remise à zéro. Cette transition remet aussi le temps et la base de temps à zéro. La temporisation est également remise à zéro si l'état de signal égale 1 à l'entrée R alors que la temporisation ne s'exécute pas.

La valeur de temps en cours peut être lue en format binaire à la sortie DUAL et en format décimal codé binaire à la sortie DEZ.

Lorsque vous placez un pavé de temporisation, vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 9-7 Temporisation sous forme de retard à la montée : représentation et paramètres, avec les abréviations SIMATIC

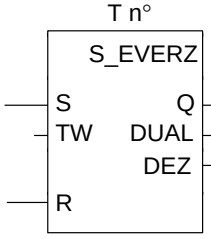
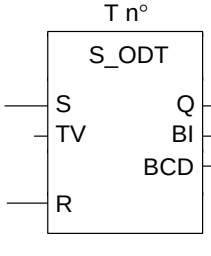
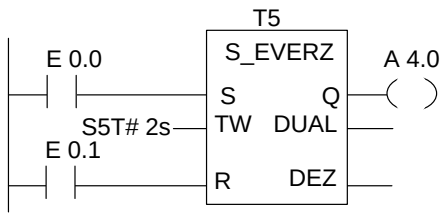
Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	n°	TIMER	T	Numéro d'identification de la temporisation. La plage dépend de la CPU.
	S	BOOL	E, A, M, D, L, T, Z	Entrée de démarrage
	TW	S5TIME	E, A, M, D, L	Valeur de temps prédéfinie (plage : 0 à 9999)
	R	BOOL	E, A, M, D, L, T, Z	Entrée de remise à zéro
	Q	BOOL	E, A, M, D, L	Etat de la temporisation
	DUAL	WORD	E, A, M, D, L	Valeur de temps restante (format binaire)
	DEZ	WORD	E, A, M, D, L	Valeur de temps restante (format DCB)

Tableau 9-8 Temporisation sous forme de retard à la montée : représentation et paramètres, avec les abréviations internationales

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	n°	TIMER	T	Numéro d'identification de la temporisation. La plage dépend de la CPU.
	S	BOOL	E, A, M, D, L, T, Z	Entrée de démarrage
	TV	S5TIME	E, A, M, D, L	Valeur de temps prédéfinie (plage : 0 à 9999)
	R	BOOL	E, A, M, D, L, T, Z	Entrée de remise à zéro
	Q	BOOL	E, A, M, D, L	Etat de la temporisation
	BI	WORD	E, A, M, D, L	Valeur de temps restante (format binaire)
	BCD	WORD	E, A, M, D, L	Valeur de temps restante (format DCB)

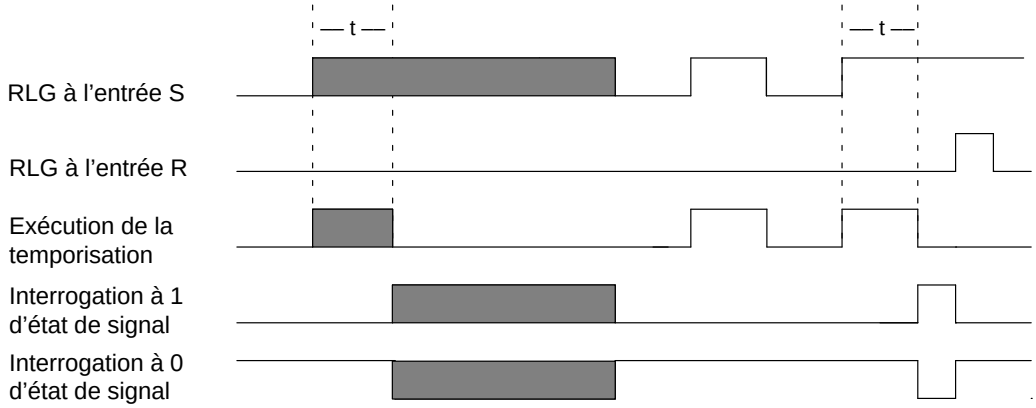


La temporisation T 5 est démarrée si l'état de signal passe de 0 à 1 à l'entrée E 0.0 (front montant du RLG). Si le temps de deux secondes (2s) indiqué expire et que l'état de signal à l'entrée E 0.0 égale toujours 1, l'état de signal à la sortie A 4.0 est 1. Si l'état de signal de l'entrée E 0.0 passe de 1 à 0, la temporisation s'arrête et A 4.0 est à 0 (voir aussi paragraphe 9.3). Si l'état de signal de l'entrée E 0.0 passe de 0 à 1 alors que la temporisation s'exécute, la temporisation est redémarrée.

Description des bits du mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	–	–	–	–	–	x	x	x	1

Chronogramme


— t —
— t —
RLG à l'entrée S
RLG à l'entrée R
Exécution de la temporisation
Interrogation à 1 d'état de signal
Interrogation à 0 d'état de signal

t = temps programmé

Figure 9-5 Temporisation sous forme de retard à la montée

9.6 Temporisation sous forme de retard à la montée mémorisé

Description

L'opération « Temporisation sous forme de retard à la montée mémorisé » démarre la temporisation précisée en cas de front montant (c'est-à-dire lorsque l'état de signal passe de 0 à 1) à l'entrée de démarrage S. Un changement d'état de signal est toujours nécessaire pour démarrer une temporisation. La valeur de temps indiquée à l'entrée TW continue à s'écouler même si l'état de signal à l'entrée S passe à 0 avant que la temporisation n'ait expiré. L'interrogation à 1 de l'état de signal à la sortie Q donne 1 comme résultat lorsque le temps a expiré, quel que soit l'état de signal à l'entrée S, et lorsque l'entrée de remise à zéro (R) reste à 0. Si l'état de signal à l'entrée S passe de 0 à 1 alors que la temporisation s'exécute, cette dernière est redémarrée avec la valeur de temps indiquée.

En cas de passage de 0 à 1 à l'entrée de remise à zéro (R), la temporisation est remise à zéro quel que soit le RLG à l'entrée S.

La valeur de temps en cours peut être lue en format binaire à la sortie DUAL et en format décimal codé binaire à la sortie DEZ.

Tableau 9-9 Temporisation sous forme de retard à la montée mémorisé : représentation et paramètres, avec les abréviations SIMATIC

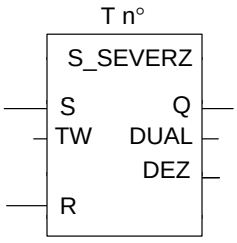
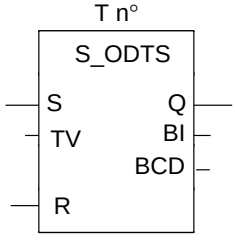
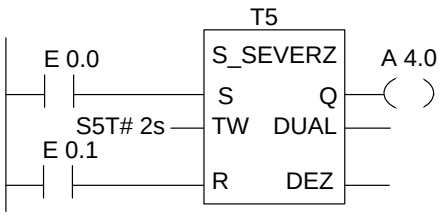
Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	n°	TIMER	T	Numéro d'identification de la temporisation. La plage dépend de la CPU.
	S	BOOL	E, A, M, D, L, T, Z	Entrée de démarrage
	TW	S5TIME	E, A, M, D, L	Valeur de temps prédéfinie (plage : 0 à 9999)
	R	BOOL	E, A, M, D, L, T, Z	Entrée de remise à zéro
	Q	BOOL	E, A, M, D, L	Etat de la temporisation
	DUAL	WORD	E, A, M, D, L	Valeur de temps restante (format binaire)
	DEZ	WORD	E, A, M, D, L	Valeur de temps restante (format DCB)

Tableau 9-10 Temporisation sous forme de retard à la montée mémorisé : représentation et paramètres, avec les abréviations internationales

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	n°	TIMER	T	Numéro d'identification de la temporisation. La plage dépend de la CPU.
	S	BOOL	E, A, M, D, L, T, Z	Entrée de démarrage
	TV	S5TIME	E, A, M, D, L	Valeur de temps prédéfinie (plage : 0 à 9999)
	R	BOOL	E, A, M, D, L, T, Z	Entrée de remise à zéro
	Q	BOOL	E, A, M, D, L	Etat de la temporisation
	BI	WORD	E, A, M, D, L	Valeur de temps restante (format binaire)
	BCD	WORD	E, A, M, D, L	Valeur de temps restante (format DCB)

Exemple

La figure 9-6 montre l'opération « Temporisation sous forme de retard à la montée mémorisé », décrit les bits du mot d'état et souligne les caractéristiques de ce type de temporisation. Lorsque vous placez un pavé de temporisation, vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

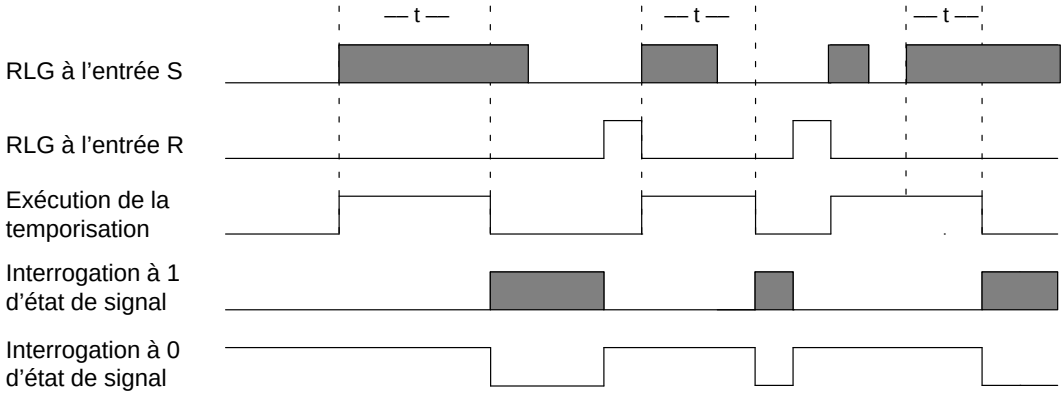


La temporisation T 5 est démarrée si l'état de signal passe de 0 à 1 à l'entrée E 0.0 (front montant du RLG). La temporisation continue à s'exécuter même si l'état de signal de l'entrée E 0.0 passe de 1 à 0. Si l'état de signal de l'entrée E 0.0 passe de 0 à 1 avant que le temps n'ait expiré, la temporisation est redémarrée. L'état de signal à la sortie A 4.0 est 1 si le temps a expiré et que E0.1 reste à 0 (voir aussi paragraphe 9.3).

Description des bits du mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	–	–	–	–	–	x	x	x	1

Chronogramme


— t —
— t —
— t —
RLG à l'entrée S
RLG à l'entrée R
Exécution de la temporisation
Interrogation à 1 d'état de signal
Interrogation à 0 d'état de signal

t = temps programmé

Figure 9-6 Temporisation sous forme de retard à la montée mémorisé

9.7 Temporisation sous forme de retard à la retombée

Description

L'opération « Temporisation sous forme de retard à la retombée » démarre la temporisation précisée en cas de front descendant (c'est-à-dire lorsque l'état de signal passe de 1 à 0) à l'entrée de démarrage S. Un changement d'état de signal est toujours nécessaire pour démarrer une temporisation. L'interrogation à 1 de l'état de signal à la sortie Q donne 1 comme résultat lorsque l'état de signal à l'entrée S est 1 ou lorsque la temporisation s'exécute. La temporisation est remise à zéro lorsque l'état de signal à l'entrée S passe de 0 à 1 alors que la temporisation s'exécute. La temporisation n'est redémarrée que lorsque l'état de signal à l'entrée S repasse de 1 à 0.

En cas de passage de 0 à 1 à l'entrée de remise à zéro (R) pendant que la temporisation s'exécute, cette dernière est remise à zéro.

La valeur de temps en cours peut être lue en format binaire à la sortie DUAL et en format décimal codé binaire à la sortie DEZ.

Lorsque vous placez un pavé de temporisation, vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 9-11 Temporisation sous forme de retard à la retombée : représentation et paramètres, avec les abréviations SIMATIC

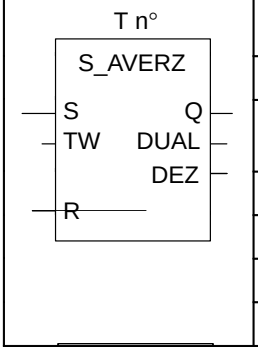
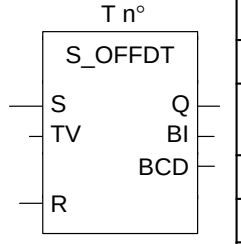
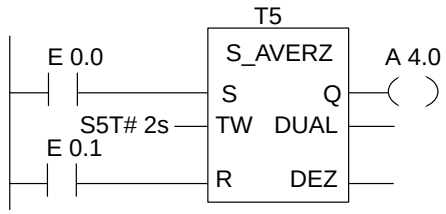
Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	n°	TIMER	T	Numéro d'identification de la temporisation. La plage dépend de la CPU.
	S	BOOL	E, A, M, D, L, T, Z	Entrée de démarrage
	TW	S5TIME	E, A, M, D, L	Valeur de temps prédéfinie (plage : 0 à 9999)
	R	BOOL	E, A, M, D, L, T, Z	Entrée de remise à zéro
	Q	BOOL	E, A, M, D, L	Etat de la temporisation
	DUAL	WORD	E, A, M, D, L	Valeur de temps restante (format binaire)
	DEZ	WORD	E, A, M, D, L	Valeur de temps restante (format DCB)

Tableau 9-12 Temporisation sous forme de retard à la retombée : représentation et paramètres, avec les abréviations internationales

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	n°	TIMER	T	Numéro d'identification de la temporisation. La plage dépend de la CPU.
	S	BOOL	E, A, M, D, L, T, Z	Entrée de démarrage
	TV	S5TIME	E, A, M, D, L	Valeur de temps prédéfinie (plage : 0 à 9999)
	R	BOOL	E, A, M, D, L, T, Z	Entrée de remise à zéro
	Q	BOOL	E, A, M, D, L	Etat de la temporisation
	BI	WORD	E, A, M, D, L	Valeur de temps restante (format binaire)
	BCD	WORD	E, A, M, D, L	Valeur de temps restante (format DCB)

Exemple

La figure 9-7 montre l'opération « Temporisation sous forme de retard à la retombée », décrit les bits du mot d'état et souligne les caractéristiques de ce type de temporisation.

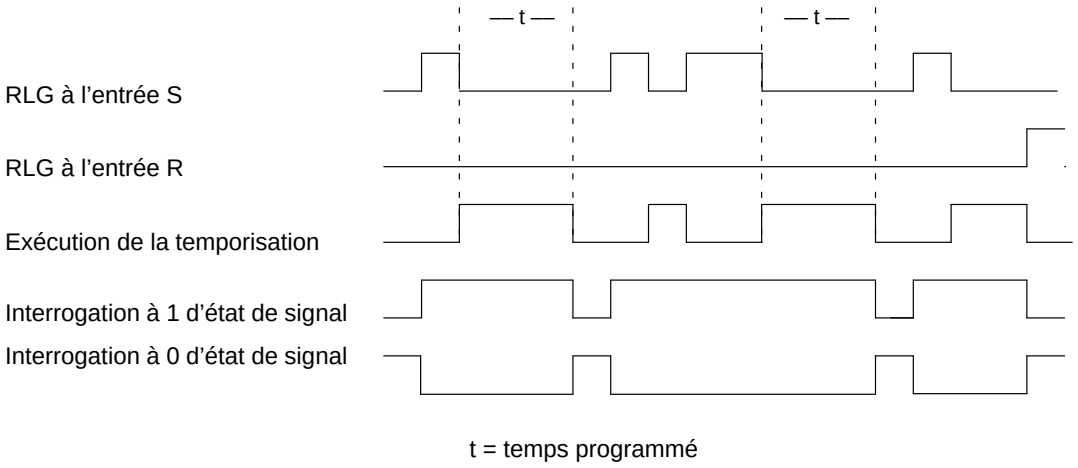


La temporisation T 5 est démarrée si l'état de signal passe de 1 à 0 à l'entrée E 0.0 (c'est-à-dire en cas de front descendant au RLG). L'état de signal à la sortie A 4.0 est 1 lorsque l'état de signal de l'entrée E 0.0 est 1 ou que la temporisation s'exécute (voir aussi paragraphe 9.3). Si l'état de signal de l'entrée E 0.1 passe de 0 à 1 alors que la temporisation s'exécute, la temporisation est remise à zéro.

Description des bits du mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	–	–	–	–	–	x	x	x	1

Chronogramme


— t —
— t —
RLG à l'entrée S
RLG à l'entrée R
Exécution de la temporisation
Interrogation à 1 d'état de signal
Interrogation à 0 d'état de signal

t = temps programmé

Figure 9-7 Temporisation sous forme de retard à la retombée

Opérations de comptage

10

Contenu de ce chapitre

Paragraphe	Thème	Page
10.1	Adresse d’un compteur en mémoire et composants d’un compteur	10-2
10.2	Compteur incrémental/décrémental	10-3
10.3	Compteur incrémental	10-5
10.4	Compteur décrémental	10-7

10.1 Adresse d'un compteur en mémoire et composants d'un compteur

Zone de mémoire

Une zone de mémoire est réservée aux compteurs dans votre CPU. Un mot de 16 bits y est réservé pour chaque compteur. La programmation en CONT permet d'utiliser jusqu'à 256 compteurs.

Les opérations de comptage sont les seules fonctions à avoir accès à la zone de mémoire réservée aux compteurs.

Valeur de comptage

La valeur de comptage est contenue dans les bits 0 à 9 du mot de comptage. Lorsque le compteur est mis à 1, la valeur que vous avez définie y est placée par l'accumulateur. La plage de la valeur de comptage est comprise entre 0 et 999. Vous pouvez modifier cette valeur en utilisant les opérations « Compteur incrémental/décrémental », « Compteur incrémental » et « Compteur décrémental ».

Configuration des bits dans le compteur

Pour assigner une valeur initiale à un compteur, vous chargez un nombre compris entre 0 et 999, par exemple 127, au format suivant comme valeur de comptage :

C# 127

C# correspond au format décimal codé binaire (format DCB : chaque groupe de quatre bits contient le code binaire d'une valeur décimale).

Les bits 0 à 11 du compteur contiennent la valeur de comptage en format DCB. La figure 10-1 montre le contenu du compteur après le chargement de la valeur 127, ainsi que le contenu de la cellule de compteur après assignation d'une valeur.

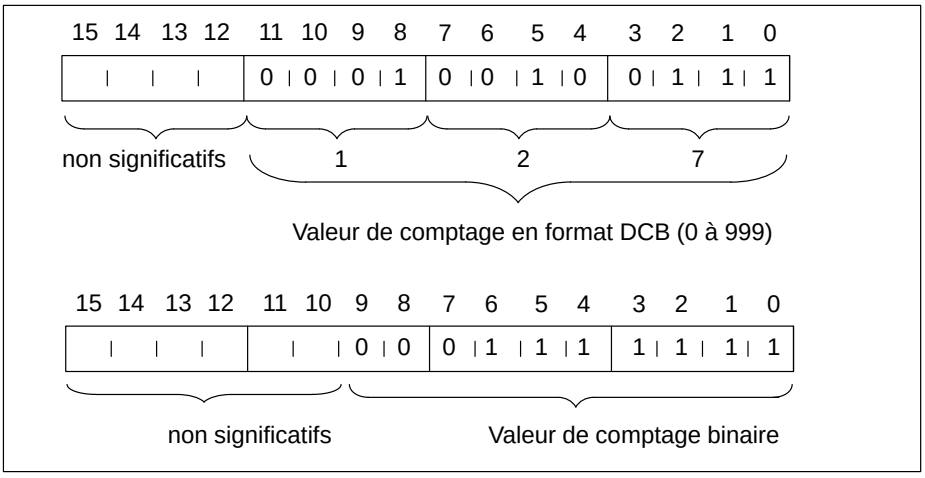

15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
0 0 0 1 0 0 1 0 0 1 1 1
non significatifs
1
2
7
Valeur de comptage en format DCB (0 à 999)
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
0 0 0 1 1 1 1 1 1 1 1
non significatifs
Valeur de comptage binaire

Figure 10-1 Contenu de la cellule de compteur après assignation de la valeur de comptage 127

10.2 Compteur incrémental/décrémental

Description

Un front montant (c'est-à-dire un changement de l'état de signal de 0 à 1) à l'entrée S de l'opération « Compteur incrémental/décrémental » initialise le compteur à la valeur figurant dans l'entrée ZW. En cas de front montant à l'entrée R, le compteur est remis à 0 et la valeur d'initialisation est mise à 0. Le compteur est incrémenté d'une unité si l'état de signal à l'entrée ZV passe de 0 à 1 – front montant – et que la valeur du compteur soit inférieure à 999. Le compteur est décrémenté d'une unité si l'état de signal à l'entrée ZR passe de 0 à 1 – front montant – et que la valeur du compteur soit supérieure à 0. En cas de front montant aux deux entrées de comptage, les deux fonctions sont exécutées et le compteur reste inchangé. L'interrogation à 1 de l'état de signal de la sortie Q donne 1 comme résultat lorsque le compteur est supérieur à 0 ; cette interrogation donne 0 comme résultat lorsque le compteur est égal à 0.

Lorsque vous placez le pavé « Compteur incrémental/décrémental », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 10-1 Compteur incrémental/décrémental : représentation et paramètres, avec les abréviations SIMATIC

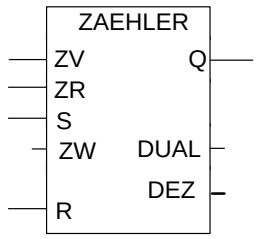
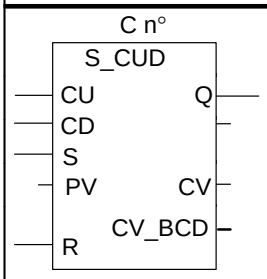
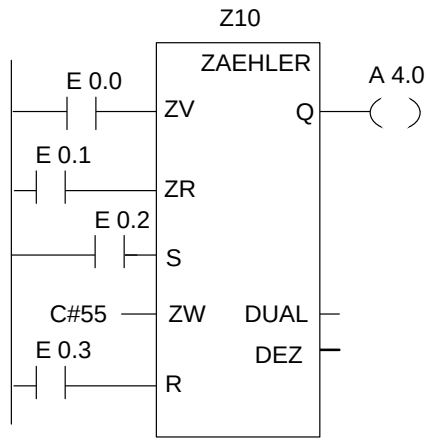
Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	n°	COUNTER	–	Numéro d'identification du compteur. La plage dépend de la CPU.
	ZV	BOOL	E, A, M, D, L	Entrée d'incrémentation (ZV)
	ZR	BOOL	E, A, M, D, L	Entrée de décrémentation (ZR)
	S	BOOL	E, A, M, D, L	Entrée d'initialisation du compteur
	ZW	WORD	E, A, M, D, L	Valeur d'initialisation du compteur comprise entre 0 et 999 (entrée sous la forme C#<valeur> pour indiquer le format DCB)
	R	BOOL	E, A, M, D, L	Entrée de remise à zéro
	Q	BOOL	E, A, M, D, L	Etat du compteur
	DUAL	WORD	E, A, M, D, L	Valeur de comptage en cours (format binaire)
	DEZ	WORD	E, A, M, D, L	Valeur de comptage en cours (format DCB)

Tableau 10-2 Compteur incrémental/décrémental : représentation et paramètres, avec les abréviations internationales

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	n°	COUNTER	–	Numéro d'identification du compteur. La plage dépend de la CPU.
	CU	BOOL	E, A, M, D, L	Entrée d'incrémentation
	CD	BOOL	E, A, M, D, L	Entrée de décrémentation
	S	BOOL	E, A, M, D, L	Entrée d'initialisation du compteur
	PV	WORD	E, A, M, D, L	Valeur d'initialisation du compteur comprise entre 0 et 999 (entrée sous la forme C#<valeur> pour indiquer le format DCB)
	R	BOOL	E, A, M, D, L	Entrée de remise à zéro
	Q	BOOL	E, A, M, D, L	Etat du compteur
	CV	WORD	E, A, M, D, L	Valeur de comptage en cours (format binaire)
	CV_BCD	WORD	E, A, M, D, L	Valeur de comptage en cours (format DCB)



Le compteur Z10 est initialisé à la valeur 55 en format DCB si l'état de signal passe de 0 à 1 à l'entrée E 0.2. Si l'état de signal de l'entrée E 0.0 passe de 0 à 1, la valeur du compteur Z10 est incrémentée d'un à moins qu'elle ne soit déjà égale à 999. Si l'état de signal de l'entrée E 0.1 passe de 0 à 1, la valeur du compteur Z10 est décrémentée d'un à moins qu'elle ne soit déjà égale à 0. Si l'état de signal de l'entrée E 0.3 passe de 0 à 1, la valeur du compteur Z10 est mise à 0. L'état de signal de la sortie A 4.0 est 1 tant que Z10 est différent de zéro.

Description des bits du mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	–	–	–	–	–	x	x	x	1

Figure 10-2 Compteur incrémental/décrémental

10.3 Compteur incrémental

Description

Un front montant (c'est-à-dire un changement de l'état de signal de 0 à 1) à l'entrée S de l'opération « Compteur incrémental » initialise le compteur à la valeur figurant dans l'entrée ZW. En cas de front montant à l'entrée R, le compteur est remis à 0 et la valeur d'initialisation est mise à 0. Le compteur est incrémenté d'une unité si l'état de signal à l'entrée ZV passe de 0 à 1 – front montant – et que la valeur du compteur soit inférieure à 999. L'interrogation à 1 de l'état de signal de la sortie Q donne 1 comme résultat lorsque le compteur est supérieur à 0 ; cette interrogation donne 0 comme résultat lorsque le compteur est égal à 0.

Lorsque vous placez le pavé « Compteur incrémental », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 10-3 Compteur incrémental : représentation et paramètres, avec les abréviations SIMATIC

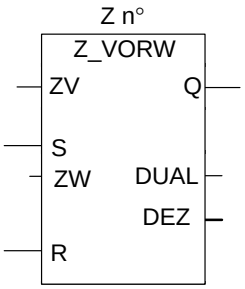
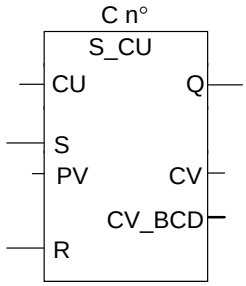
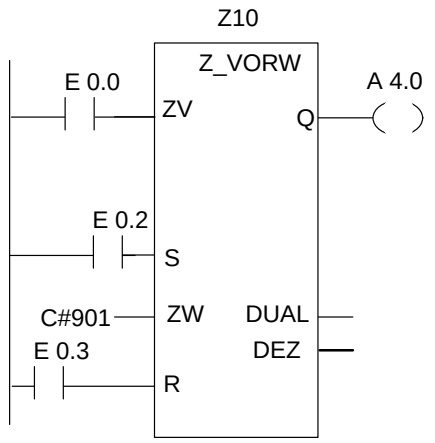
Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	n°	COUNTER	Z	Numéro d'identification du compteur. La plage dépend de la CPU.
	ZV	BOOL	E, A, M, D, L	Entrée d'incrémentation (ZV)
	S	BOOL	E, A, M, D, L	Entrée d'initialisation du compteur
	ZW	WORD	E, A, M, D, L	Valeur d'initialisation du compteur comprise entre 0 et 999 (entrée sous la forme C#<valeur> pour indiquer le format DCB)
	R	BOOL	E, A, M, D, L	Entrée de remise à zéro
	Q	BOOL	E, A, M, D, L	Etat du compteur
	DUAL	WORD	E, A, M, D, L	Valeur de comptage en cours (format binaire)
	DEZ	WORD	E, A, M, D, L	Valeur de comptage en cours (format DCB)

Tableau 10-4 Compteur incrémental : représentation et paramètres, avec les abréviations internationales

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	n°	COUNTER	–	Numéro d'identification du compteur. La plage dépend de la CPU.
	CU	BOOL	E, A, M, D, L	Entrée d'incrémentation
	S	BOOL	E, A, M, D, L	Entrée d'initialisation du compteur
	PV	WORD	E, A, M, D, L	Valeur d'initialisation du compteur comprise entre 0 et 999 (entrée sous la forme C#<valeur> pour indiquer le format DCB)
	R	BOOL	E, A, M, D, L	Entrée de remise à zéro
	Q	BOOL	E, A, M, D, L	Etat du compteur
	CV	WORD	E, A, M, D, L	Valeur de comptage en cours (format binaire)
	CV_BCD	WORD	E, A, M, D, L	Valeur de comptage en cours (format DCB)



Le compteur Z10 est initialisé à la valeur 901 en format DCB si l'état de signal passe de 0 à 1 à l'entrée E 0.2. Si l'état de signal en E 0.0 passe de 0 à 1, la valeur du compteur Z10 est incrémentée d'un à moins qu'elle ne soit déjà égale à 999. Si l'état de signal en E 0.3 passe de 0 à 1, la valeur du compteur Z10 est mise à 0. L'état de signal de la sortie A 4.0 est 1 si Z10 est différent de zéro.

Description des bits du mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	–	–	–	–	–	x	x	x	1

Figure 10-3 Compteur incrémental

10.4 Compteur décrémental

Description

Un front montant (c'est-à-dire un changement de l'état de signal de 0 à 1) à l'entrée S de l'opération « Compteur décrémental » initialise le compteur à la valeur figurant dans l'entrée ZW. En cas de front montant à l'entrée R, le compteur est remis à 0, et la valeur d'initialisation est mise à 0. Le compteur est décrémenté d'une unité si l'état de signal à l'entrée ZR passe de 0 à 1 – front montant – et que la valeur du compteur soit supérieure à 0. L'interrogation à 1 de l'état de signal de la sortie Q donne 1 comme résultat lorsque le compteur est supérieur à 0 ; cette interrogation donne 0 comme résultat lorsque le compteur est égal à 0.

Lorsque vous placez le pavé « Compteur décrémental », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 10-5 Compteur décrémental : représentation et paramètres, avec les abréviations SIMATIC

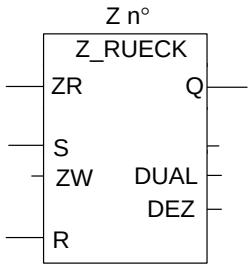
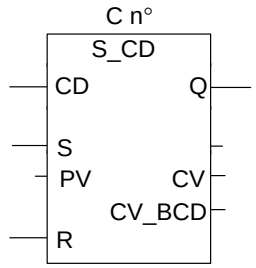
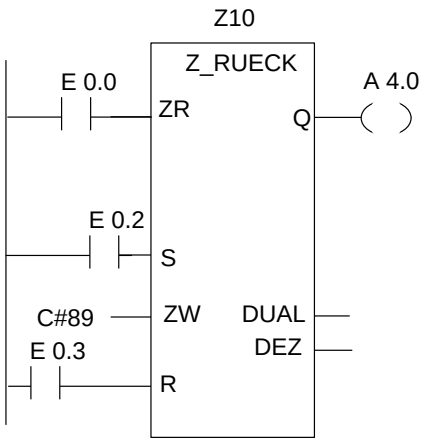
Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	n°	COUNTER	Z	Numéro d'identification du compteur. La plage dépend de la CPU.
	ZR	BOOL	E, A, M, D, L	Entrée de décrémentation (ZR)
	S	BOOL	E, A, M, D, L	Entrée d'initialisation du compteur
	ZW	WORD	E, A, M, D, L	Valeur d'initialisation du compteur comprise entre 0 et 999 (entrée sous la forme C#<valeur> pour indiquer le format DCB)
	R	BOOL	E, A, M, D, L	Entrée de remise à zéro
	Q	BOOL	E, A, M, D, L	Etat du compteur
	DUAL	WORD	E, A, M, D, L	Valeur de comptage en cours (format binaire)
	DEZ	WORD	E, A, M, D, L	Valeur de comptage en cours (format DCB)

Tableau 10-6 Compteur décrémental : représentation et paramètres, avec les abréviations internationales

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	n°	COUNTER	–	Numéro d'identification du compteur. La plage dépend de la CPU.
	CD	BOOL	E, A, M, D, L	Entrée de décrémentation (ZR)
	S	BOOL	E, A, M, D, L	Entrée d'initialisation du compteur
	PV	WORD	E, A, M, D, L	Valeur d'initialisation du compteur comprise entre 0 et 999 (entrée sous la forme C#<valeur> pour indiquer le format DCB)
	R	BOOL	E, A, M, D, L	Entrée de remise à zéro
	Q	BOOL	E, A, M, D, L	Etat du compteur
	CV	WORD	E, A, M, D, L	Valeur de comptage en cours (format binaire)
	CV_BCD	WORD	E, A, M, D, L	Valeur de comptage en cours (format DCB)



Le compteur Z10 est initialisé à la valeur 89 en format DCB si l'état de signal passe de 0 à 1 à l'entrée E 0.2. Si l'état de signal de l'entrée E 0.0 passe de 0 à 1, la valeur du compteur Z10 est décrémentée d'un à moins qu'elle ne soit déjà égale à 0. Si l'état de signal de l'entrée E 0.3 passe de 0 à 1, la valeur du compteur Z10 est mise à zéro.

Description des bits du mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	–	–	–	–	–	x	x	x	1

Figure 10-4 Compteur décrémental

Opérations arithmétiques sur nombres entiers

11

Contenu de ce chapitre

Paragraphe	Thème	Page
11.1	Additionner entiers de 16 bits	11-2
11.2	Additionner entiers de 32 bits	11-3
11.3	Soustraire entiers de 16 bits	11-4
11.4	Soustraire entiers de 32 bits	11-5
11.5	Multiplier entiers de 16 bits	11-6
11.6	Multiplier entiers de 32 bits	11-7
11.7	Diviser entiers de 16 bits	11-8
11.8	Diviser entiers de 32 bits	11-9
11.9	Reste de division (32 bits)	11-10
11.10	Evaluation des bits du mot d'état pour les opérations arithmétiques sur nombres entiers	11-11

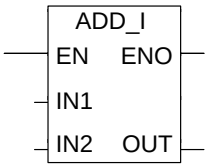
11.1 Additionner entiers de 16 bits

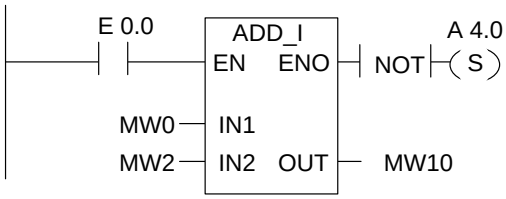
Description

L'opération « Additionner entiers de 16 bits » est activée si l'état de signal est 1 à l'entrée de validation EN. Cette opération additionne les entrées IN1 et IN2 et range le résultat dans la sortie OUT. Si ce résultat est hors de la plage autorisée pour un nombre entier de 16 bits, les bits de débordement et de débordement mémorisé (bits DEB et DM) sont mis à 1 et la sortie ENO est mise à 0.

Lorsque vous placez le pavé « Additionner entiers de 16 bits », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 11-1 Additionner entiers de 16 bits : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN1	INT	E, A, M, D, L	Première valeur pour l'addition
	IN2	INT	E, A, M, D, L	Seconde valeur pour l'addition
	OUT	INT	E, A, M, D, L	Résultat de l'addition



L'opération ADD_I est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le résultat de l'addition MW0 + MW2 est rangé dans le mot de mémento MW10. Si ce résultat est hors de la plage autorisée pour un nombre entier de 16 bits ou si l'état de signal de l'entrée E 0.0 est égal à 0, la sortie A 4.0 est mise à 1.

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	x	x	x	x	0	x	x	1

Figure 11-1 Additionner entiers de 16 bits

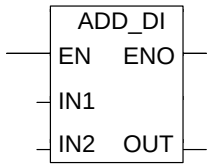
11.2 Additionner entiers de 32 bits

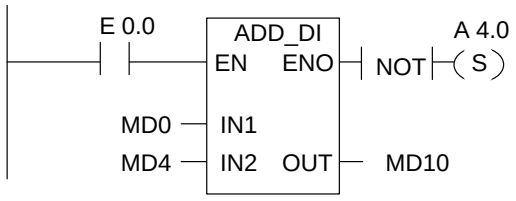
Description

L'opération « Additionner entiers de 32 bits » est activée si l'état de signal est 1 à l'entrée de validation EN. Cette opération additionne les entrées IN1 et IN2 et range le résultat dans la sortie OUT. Si ce résultat est hors de la plage autorisée pour un nombre entier de 32 bits, les bits de débordement et de débordement mémorisé (bits DEB et DM) sont mis à 1 et la sortie ENO est mise à 0.

Lorsque vous placez le pavé « Additionner entiers de 32 bits », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 11-2 Additionner entiers de 32 bits : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN1	DINT	E, A, M, D, L	Première valeur pour l'addition
	IN2	DINT	E, A, M, D, L	Seconde valeur pour l'addition
	OUT	DINT	E, A, M, D, L	Résultat de l'addition



L'opération ADD_DI est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le résultat de l'addition MD0 + MD4 est rangé dans le double mot de mémento MD10. Si ce résultat est hors de la plage autorisée pour un nombre entier de 32 bits ou si l'état de signal de l'entrée E 0.0 est égal à 0, la sortie A 4.0 est mise à 1.

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	x	x	x	x	0	x	x	1

Figure 11-2 Additionner entiers de 32 bits

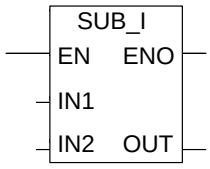
11.3 Soustraire entiers de 16 bits

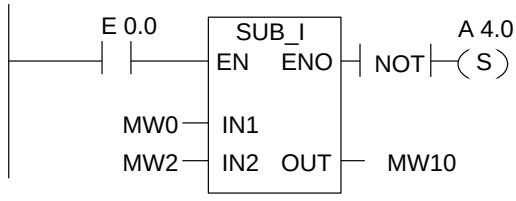
Description

L'opération « Soustraire entiers de 16 bits » est activée si l'état de signal est 1 à l'entrée de validation EN. Cette opération soustrait l'entrée IN2 de l'entrée IN1 et range le résultat dans la sortie OUT. Si ce résultat est hors de la plage autorisée pour un nombre entier de 16 bits, les bits de débordement et de débordement mémorisé (bits DEB et DM) sont mis à 1 et la sortie ENO est mise à 0.

Lorsque vous placez le pavé « Soustraire entiers de 16 bits », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 11-3 Soustraire entiers de 16 bits : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN1	INT	E, A, M, D, L	Première valeur (de laquelle soustraire)
	IN2	INT	E, A, M, D, L	Valeur à soustraire de la première valeur
	OUT	INT	E, A, M, D, L	Résultat de la soustraction



L'opération SUB_I est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le résultat de la soustraction MW0 moins MW2 est rangé dans le mot de mémento MW10. Si ce résultat est hors de la plage autorisée pour un nombre entier de 16 bits ou si l'état de signal de l'entrée E 0.0 est égal à 0, la sortie A 4.0 est mise à 1.

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	x	x	x	x	0	x	x	1

Figure 11-3 Soustraire entiers de 16 bits

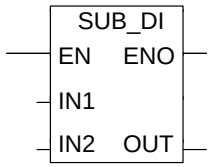
11.4 Soustraire entiers de 32 bits

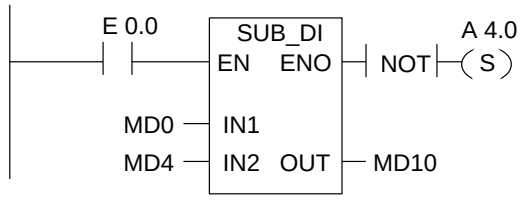
Description

L'opération « Soustraire entiers de 32 bits » est activée si l'état de signal est 1 à l'entrée de validation EN. Cette opération soustrait l'entrée IN2 de l'entrée IN1 et range le résultat dans la sortie OUT. Si ce résultat est hors de la plage autorisée pour un nombre entier de 32 bits, les bits de débordement et de débordement mémorisé (bits DEB et DM) sont mis à 1 et la sortie ENO est mise à 0.

Lorsque vous placez le pavé « Soustraire entiers de 32 bits », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 11-4 Soustraire entiers de 32 bits : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN1	DINT	E, A, M, D, L	Première valeur (de laquelle soustraire)
	IN2	DINT	E, A, M, D, L	Valeur à soustraire de la première valeur
	OUT	DINT	E, A, M, D, L	Résultat de la soustraction



L'opération SUB_DI est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le résultat de la soustraction MD0 moins MD4 est rangé dans le double mot de mémento MD10. Si ce résultat est hors de la plage autorisée pour un nombre entier de 32 bits ou si l'état de signal de l'entrée E 0.0 est égal à 0, la sortie A 4.0 est mise à 1.

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	x	x	x	x	0	x	x	1

Figure 11-4 Soustraire entiers de 32 bits

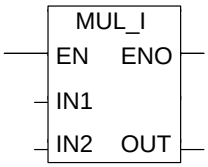
11.5 Multiplier entiers de 16 bits

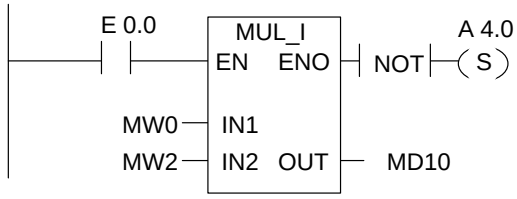
Description

L'opération « Multiplier entiers de 16 bits » est activée si l'état de signal est 1 à l'entrée de validation EN. Cette opération multiplie les entrées IN1 et IN2 et range le résultat dans la sortie OUT. Si ce résultat est hors de la plage autorisée pour un nombre entier de 16 bits, les bits de débordement et de débordement mémorisé (bits DEB et DM) sont mis à 1 et la sortie ENO est mise à 0.

Lorsque vous placez le pavé « Multiplier entiers de 16 bits », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 11-5 Multiplier entiers de 16 bits : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN1	INT	E, A, M, D, L	Première valeur pour la multiplication
	IN2	INT	E, A, M, D, L	Seconde valeur pour la multiplication
	OUT	DINT	E, A, M, D, L	Résultat de la multiplication



L'opération MUL_I est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le résultat de la multiplication MW0 x MW2 est rangé dans le double mot de mémento MD10. Si ce résultat est hors de la plage autorisée pour un nombre entier de 16 bits ou si l'état de signal de l'entrée E 0.0 est égal à 0, la sortie A 4.0 est mise à 1.

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	x	x	x	x	0	x	x	1

Figure 11-5 Multiplier entiers de 16 bits

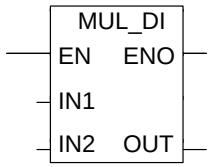
11.6 Multiplier entiers de 32 bits

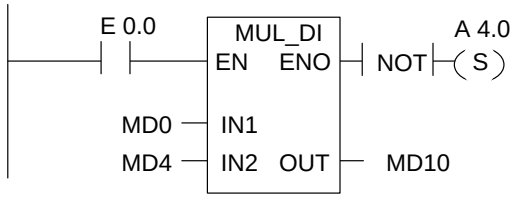
Description

L'opération « Multiplier entiers de 32 bits » est activée si l'état de signal est 1 à l'entrée de validation EN. Cette opération multiplie les entrées IN1 et IN2 et range le résultat dans la sortie OUT. Si ce résultat est hors de la plage autorisée pour un nombre entier de 32 bits, les bits de débordement et de débordement mémorisé (bits DEB et DM) sont mis à 1 et la sortie ENO est mise à 0.

Lorsque vous placez le pavé « Multiplier entiers de 32 bits », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 11-6 Multiplier entiers de 32 bits : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN1	DINT	E, A, M, D, L	Première valeur pour la multiplication
	IN2	DINT	E, A, M, D, L	Seconde valeur pour la multiplication
	OUT	DINT	E, A, M, D, L	Résultat de la multiplication



L'opération MUL_DI est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le résultat de la multiplication MD0 x MD4 est rangé dans le double mot de mémento MD10. Si ce résultat est hors de la plage autorisée pour un nombre entier de 32 bits ou si l'état de signal de l'entrée E 0.0 est égal à 0, la sortie A 4.0 est mise à 1.

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	x	x	x	x	0	x	x	1

Figure 11-6 Multiplier entiers de 32 bits

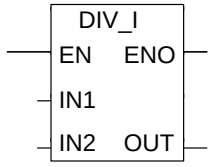
11.7 Diviser entiers de 16 bits

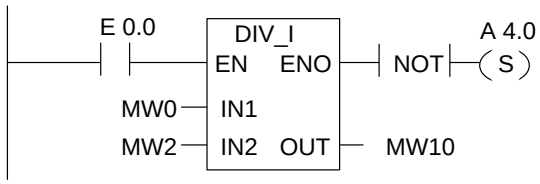
Description

L'opération « Diviser entiers de 16 bits » est activée si l'état de signal est 1 à l'entrée de validation EN. Cette opération divise l'entrée IN1 par l'entrée IN2 et range le quotient entier (résultat tronqué) dans la sortie OUT. Elle ne fournit pas de reste. Si le quotient est hors de la plage autorisée pour un nombre entier de 16 bits, les bits de débordement et de débordement mémorisé (bits DEB et DM) sont mis à 1 et la sortie ENO est mise à 0.

Lorsque vous placez le pavé « Diviser entiers de 16 bits », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 11-7 Diviser entiers de 16 bits : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN1	INT	E, A, M, D, L	Dividende
	IN2	INT	E, A, M, D, L	Diviseur
	OUT	INT	E, A, M, D, L	Résultat de la division



L'opération DIV_I est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le quotient de la division de MW0 par MW2 est rangé dans le mot de mémento MW10. Si ce quotient est hors de la plage autorisée pour un nombre entier de 16 bits ou si l'état de signal de l'entrée E 0.0 est égal à 0, la sortie A 4.0 est mise à 1.

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	x	x	x	x	0	x	x	1

Figure 11-7 Diviser entiers de 16 bits

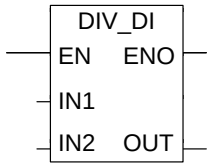
11.8 Diviser entiers de 32 bits

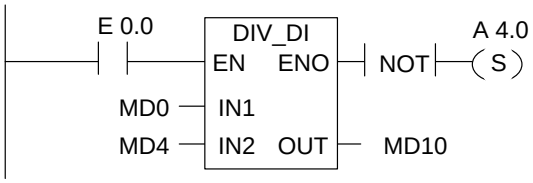
Description

L'opération « Diviser entiers de 32 bits » est activée si l'état de signal est 1 à l'entrée de validation EN. Cette opération divise l'entrée IN1 par l'entrée IN2 et range le quotient (résultat tronqué) dans la sortie OUT. Elle range le quotient sous forme de valeur unique de 32 bits en format DINT et ne fournit pas de reste. Si le quotient est hors de la plage autorisée pour un nombre entier de 32 bits, les bits de débordement et de débordement mémorisé (bits DEB et DM) sont mis à 1 et la sortie ENO est mise à 0.

Lorsque vous placez le pavé « Diviser entiers de 32 bits », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 11-8 Diviser entiers de 32 bits : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN1	DINT	E, A, M, D, L	Dividende
	IN2	DINT	E, A, M, D, L	Diviseur
	OUT	DINT	E, A, M, D, L	Résultat de la division



L'opération DIV_DI est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le quotient de la division de MD0 par MD4 est rangé dans le double mot de mémento MD10. Si ce quotient est hors de la plage autorisée pour un nombre entier de 32 bits ou si l'état de signal de l'entrée E 0.0 est égal à 0, la sortie A 4.0 est mise à 1.

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	x	x	x	x	0	x	x	1

Figure 11-8 Diviser entiers de 32 bits

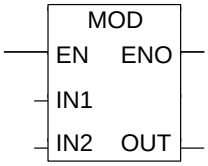
11.9 Reste de division (32 bits)

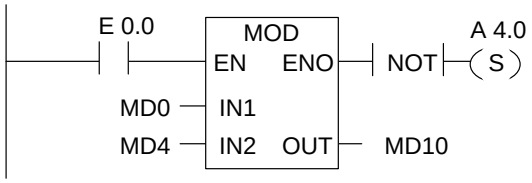
Description

L'opération « Reste de division (32 bits) » est activée si l'état de signal est 1 à l'entrée de validation EN. Cette opération divise l'entrée IN1 par l'entrée IN2 et range le reste dans la sortie OUT. Si ce reste est hors de la plage autorisée pour un nombre entier de 32 bits, les bits de débordement et de débordement mémorisé (bits DEB et DM) sont mis à 1 et la sortie ENO est mise à 0.

Lorsque vous placez le pavé « Reste de division (32 bits) », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 11-9 Reste de division (32 bits) : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN1	DINT	E, A, M, D, L	Dividende
	IN2	DINT	E, A, M, D, L	Diviseur
	OUT	DINT	E, A, M, D, L	Reste



L'opération MOD est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le reste de la division de MD0 par MD4 est rangé dans le double mot de mémento MD10. Si ce reste est hors de la plage autorisée pour un nombre entier de 32 bits ou si l'état de signal de l'entrée E 0.0 est égal à 0, la sortie A 4.0 est mise à 1.

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	x	x	x	x	0	x	x	1

Figure 11-9 Reste de division (32 bits)

11.10 Evaluation des bits du mot d'état pour les opérations arithmétiques sur nombres entiers

Les opérations arithmétiques fondamentales affectent les bits suivants du mot d'état :

- BI1 et BI0
- DEB
- DM

Un tiret (-) dans la colonne de l'un des bits du tableau signifie que le bit correspondant n'est pas affecté par le résultat de l'opération arithmétique.

Tableau 11-10 Etat de signal des bits du mot d'état pour les résultats d'opérations arithmétiques, compris dans la plage autorisée

Plage autorisée pour le résultat d'une opération sur nombres entiers (16 et 32 bits)	**Bits du mot d'état**			
	BI1	**BI0**	**DEB**	**DM**
0 (zéro)	0	0	0	-
16 bits : -32 768 ≤ résultat < 0 (nombre négatif) 32 bits : -2 147 483 648 ≤ résultat < 0 (nombre négatif)	0	1	0	-
16 bits : 32 767 ≥ résultat > 0 (nombre positif) 32 bits : 2 147 483 647 ≥ résultat > 0 (nombre positif)	1	0	0	-

Tableau 11-11 Etat de signal des bits du mot d'état pour les résultats d'opérations arithmétiques, non compris dans la plage autorisée

Plage non autorisée pour le résultat d'une opération sur nombres réels (16 et 32 bits)	**Bits du mot d'état**			
	BI1	**BI0**	**DEB**	**DM**
16 bits : résultat > 32 767 (nombre positif) 32 bits : résultat > 2 147 483 647 (nombre positif)	1	0	1	1
16 bits : résultat < -32 768 (nombre négatif) 32 bits : résultat < -2 147 483 648 (nombre négatif)	0	1	1	1

Tableau 11-12 Etat de signal des bits du mot d'état pour les opérations arithmétiques sur les nombres entiers (32 bits) +D, /D et MOD

Opération	**Bits du mot d'état**			
	BI1	**BI0**	**DEB**	**DM**
+D : résultat = -4 294 967 296	0	0	1	1
/D ou MOD : division par 0 (zéro)	1	1	1	1

Opérations arithmétiques sur nombres réels 12

Contenu de ce chapitre

Paragraphe	Thème	Page
12.1	Présentation	12-2
12.2	Additionner nombres réels	12-3
12.3	Soustraire nombres réels	12-4
12.4	Multiplier nombres réels	12-5
12.5	Diviser nombres réels	12-6
12.6	Evaluation des bits du mot d'état pour les opérations arithmétiques sur nombres réels	12-7
12.7	Valeur absolue d'un nombre réel	12-8
12.8	Carré ou racine carrée d'un nombre réel	12-9
12.9	Logarithme naturel d'un nombre réel	12-11
12.10	Valeur exponentielle d'un nombre réel	12-12
12.11	Fonctions trigonométriques d'angles sous forme de nombres réels	12-13

12.1 Présentation

Les opérations arithmétiques sur nombres réels permettent d'exécuter les fonctions arithmétiques suivantes sur **deux** nombres à virgule flottante IEEE de 32 bits :

- addition,
- soustraction,
- multiplication,
- division.

Les nombres à virgule flottante IEEE de 32 bits ont le type de données REAL. Pour obtenir des informations sur le format des nombres réels, voir l'annexe C.

Les opérations arithmétiques sur nombres réels permettent d'exécuter les fonctions arithmétiques suivantes sur **un** nombre à virgule flottante IEEE de 32 bits :

- Carré (SQR) ou racine carrée (SQRT) d'un nombre réel
- Logarithme naturel (LN) d'un nombre réel
- Valeur exponentielle d'un nombre réel (EXP) sur la base e (= 2,71828...)
- Fonctions trigonométriques d'angles représentés sous forme de nombres réels IEEE de 32 bits :
  - Fonctions sinus d'un nombre réel (SIN) et arc sinus d'un nombre réel (ASIN)
  - Fonctions cosinus d'un nombre réel (COS) et arc cosinus d'un nombre réel (ACOS)
  - Fonctions tangente d'un nombre réel (TAN) et arc tangente d'un nombre réel (ATAN)

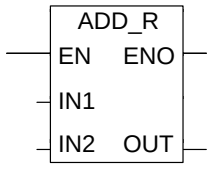
12.2 Additionner nombres réels

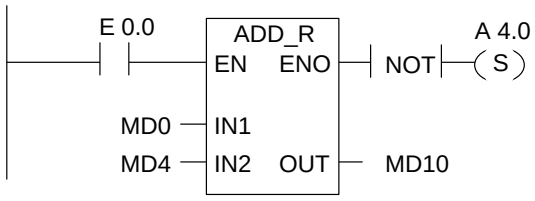
Description

L'opération « Additionner nombres réels » est activée si l'état de signal est 1 à l'entrée de validation EN. Cette opération additionne les entrées IN1 et IN2 et range le résultat dans la sortie OUT. Si ce résultat est hors de la plage autorisée pour un nombre réel (dépassement haut ou bas), les bits de débordement et de débordement mémorisé (bits DEB et DM) sont mis à 1 et la sortie ENO est mise à 0. Pour des informations sur l'évaluation des bits du mot d'état, reportez-vous au paragraphe 12.6.

Lorsque vous placez le pavé « Additionner nombres réels », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 12-1 Additionner nombres réels : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN1	REAL	E, A, M, D, L	Première valeur pour l'addition
	IN2	REAL	E, A, M, D, L	Seconde valeur pour l'addition
	OUT	REAL	E, A, M, D, L	Résultat de l'addition



L'opération ADD_R est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le résultat de l'addition MD0 + MD4 est rangé dans le double mot de mémento MD10. Si ce résultat est hors de la plage autorisée pour un nombre réel ou si l'état de signal de l'entrée E 0.0 est égal à 0, la sortie A 4.0 est mise à 1.

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	x	x	x	x	0	x	x	1

Figure 12-1 Additionner nombres réels

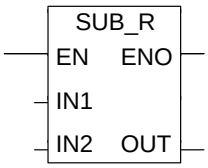
12.3 Soustraire nombres réels

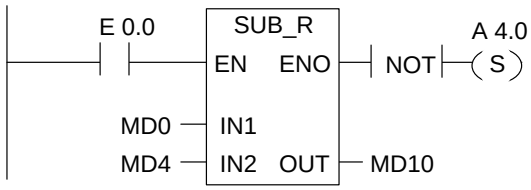
Description

L'opération « Soustraire nombres réels » est activée si l'état de signal est 1 à l'entrée de validation EN. Cette opération soustrait l'entrée IN2 de l'entrée IN1 et range le résultat dans la sortie OUT. Si ce résultat est hors de la plage autorisée pour un nombre réel (dépassement haut ou bas), les bits de débordement et de débordement mémorisé (bits DEB et DM) sont mis à 1 et la sortie ENO est mise à 0. Pour des informations sur l'évaluation des bits du mot d'état, reportez-vous au paragraphe 12.6.

Lorsque vous placez le pavé « Soustraire nombres réels », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 12-2 Soustraire nombres réels : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN1	REAL	E, A, M, D, L	Première valeur (de laquelle soustraire)
	IN2	REAL	E, A, M, D, L	Valeur à soustraire de la première valeur
	OUT	REAL	E, A, M, D, L	Résultat de la soustraction



L'opération SUB_R est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le résultat de la soustraction MD0 - MD4 est rangé dans le double mot de mémento MD10. Si ce résultat est hors de la plage autorisée pour un nombre réel ou si l'état de signal de l'entrée E 0.0 est égal à 0, la sortie A 4.0 est mise à 1.

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	x	x	x	x	0	x	x	1

Figure 12-2 Soustraire nombres réels

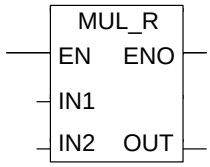
12.4 Multiplier nombres réels

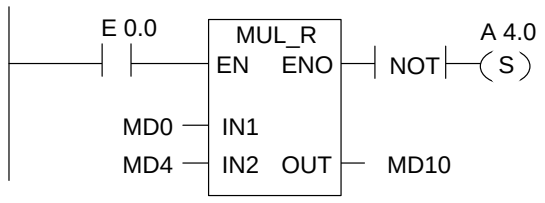
Description

L'opération « Multiplier nombres réels » est activée si l'état de signal est 1 à l'entrée de validation EN. Cette opération multiplie les entrées IN1 et IN2 et range le résultat dans la sortie OUT. Si ce résultat est hors de la plage autorisée pour un nombre réel (dépassement haut ou bas), les bits de débordement et de débordement mémorisé (bits DEB et DM) sont mis à 1 et la sortie ENO est mise à 0. Pour des informations sur l'évaluation des bits du mot d'état, reportez-vous au paragraphe 12.6.

Lorsque vous placez le pavé « Multiplier nombres réels », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 12-3 Multiplier nombres réels : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN1	REAL	E, A, M, D, L	Première valeur pour la multiplication
	IN2	REAL	E, A, M, D, L	Seconde valeur pour la multiplication
	OUT	REAL	E, A, M, D, L	Résultat de la multiplication



L'opération MUL_R est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le résultat de la multiplication MD0 x MD4 est rangé dans le double mot de mémento MD10. Si ce résultat est hors de la plage autorisée pour un nombre réel ou si l'état de signal de l'entrée E 0.0 est égal à 0, la sortie A 4.0 est mise à 1.

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	x	x	x	x	0	x	x	1

Figure 12-3 Multiplier nombres réels

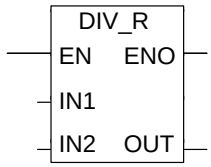
12.5 Diviser nombres réels

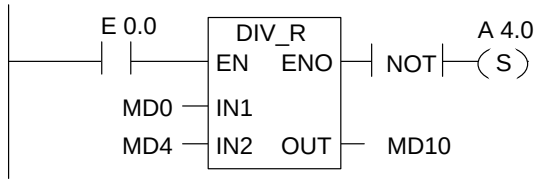
Description

L'opération « Diviser nombres réels » est activée si l'état de signal est 1 à l'entrée de validation EN. Cette opération divise l'entrée IN1 par l'entrée IN2 et range le résultat dans la sortie OUT. Si ce résultat est hors de la plage autorisée pour un nombre réel (dépassement haut ou bas), les bits de débordement et de débordement mémorisé (bits DEB et DM) sont mis à 1 et la sortie ENO est mise à 0. Pour des informations sur l'évaluation des bits du mot d'état, reportez-vous au paragraphe 12.6.

Lorsque vous placez le pavé « Diviser nombres réels », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 12-4 Diviser nombres réels : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN1	REAL	E, A, M, D, L	Dividende
	IN2	REAL	E, A, M, D, L	Diviseur
	OUT	REAL	E, A, M, D, L	Résultat de la division



L'opération DIV_R est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le résultat de la division de MD0 par MD4 est rangé dans le double mot de mémento MD10. Si ce résultat est hors de la plage autorisée pour un nombre réel ou si l'état de signal de l'entrée E0.0 est égal à 0, la sortie A 4.0 est mise à 1.

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	x	x	x	x	0	x	x	1

Figure 12-4 Diviser nombres réels

12.6 Evaluation des bits du mot d'état pour les opérations arithmétiques sur nombres réels

Description

Les opérations arithmétiques fondamentales affectent les bits suivants du mot d'état :

- BI1 et BI0
- DEB
- DM

Un tiret (-) dans la colonne de l'un des bits du tableau signifie que le bit correspondant n'est pas affecté par le résultat de l'opération arithmétique.

Tableau 12-5 Etat de signal des bits du mot d'état pour les résultats d'opérations arithmétiques sur nombres réels, compris dans la plage autorisée

Plage autorisée pour le résultat d'une opération sur nombres réels (32 bits)	Bits du mot d'état			
	BI1	BI0	DEB	DM
+0, -0 (zéro)	0	0	0	-
-3,402823E+38 < résultat < -1,175494E-38 (nombre négatif)	0	1	0	-
+1,175494E-38 < résultat < 3,402823E+38 (nombre positif)	1	0	0	-

Tableau 12-6 Etat de signal des bits du mot d'état pour les résultats d'opérations arithmétiques sur nombres réels, non compris dans la plage autorisée

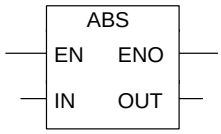
Plage non autorisée pour le résultat d'une opération sur nombres réels (32 bits)	Bits du mot d'état			
	BI1	BI0	DEB	DM
-1,175494E-38 < résultat < - 1,401298E-45 (nombre négatif) dépassement bas	0	0	1	1
+1,401298E-45 < résultat < +1,175494E-38 (nombre positif) dépassement bas	0	0	1	1
Résultat < -3,402823E+38 (nombre négatif) débordement	0	1	1	1
Résultat > 3,402823E+38 (nombre positif) débordement	1	0	1	1
Résultat < -3,402823E+38 ou résultat > +3,402823E+38 Pas un nombre réel	1	1	1	1

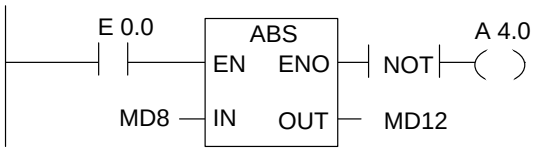
12.7 Valeur absolue d’un nombre réel

Description

L’opération « Valeur absolue d’un nombre réel » permet d’obtenir la valeur absolue d’un nombre réel.

Tableau 12-7 Valeur absolue d’un nombre réel : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN	REAL	E, A, M, D, L	Valeur d’entrée : nombre réel
	OUT	REAL	E, A, M, D, L	Valeur de sortie : valeur absolue du nombre réel



Si E 0.0 est à 1, la valeur absolue du nombre réel figurant dans MD8 est transmise dans MD12.

MD8 = +6,234 x 10^{-3} donne
MD12 = 6,234 x 10^{-3}.

La sortie A 4.0 est mise à 1 si la conversion n’est pas exécutée (ENO = EN = 0).

Description des bits du mot d’état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	X	–	–	–	–	0	X	X	1

Figure 12-5 Valeur absolue d’un nombre réel

12.8 Carré ou racine carrée d'un nombre réel

Description

L'opération « Carré d'un nombre réel » vous permet de constituer le carré d'un nombre réel.

L'opération « Racine carrée d'un nombre réel » vous permet de constituer la racine carrée d'un nombre réel. Cette opération fournit un résultat positif lorsque l'opérande est supérieur à 0, à la seule exception de la racine carrée de 0 qui est égale à 0.

Le paragraphe 12.6 fournit des informations sur la façon dont les opérations SQR ou SQRT affectent l'état de signal des bits d'état BI1, BI0, DEB et DM.

Paramètres

Le tableau 12-8 représente le pavé SQR et décrit ses paramètres. Le tableau 12-9 représente le pavé SQRT et décrit ses paramètres.

Tableau 12-8 Carrée d'un nombre réel : représentation et paramètres

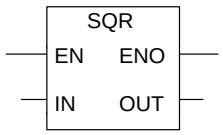
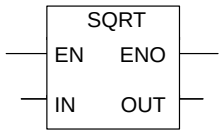
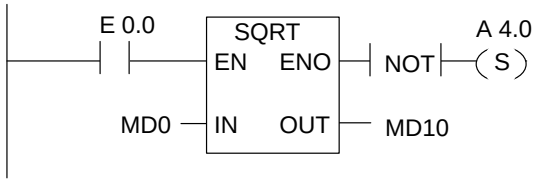
Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN	REAL	E, A, M, D, L	Nombre réel
	OUT	REAL	E, A, M, D, L	Carré du nombre réel

Tableau 12-9 Racine carrée d'un nombre réel : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN	REAL	E, A, M, D, L	Nombre réel
	OUT	REAL	E, A, M, D, L	Racine carrée du nombre réel



L'opération SQRT est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le résultat de SQRT (MD0) est rangé dans le double mot de mémento MD10. Si MD0 < 0 ou si le résultat est hors de la plage autorisée pour un nombre réel ou si l'état de signal de l'entrée E0.0 est égal à 0, la sortie A 4.0 est mise à 1.

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	x	x	x	x	0	x	x	1

Figure 12-6 Racine carrée d'un nombre réel

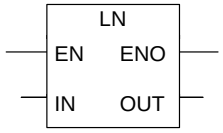
12.9 Logarithme naturel d'un nombre réel

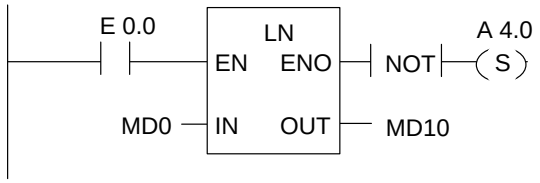
Description

L'opération « Logarithme naturel d'un nombre réel » vous permet de constituer le logarithme naturel d'un nombre réel.

Le paragraphe 12.6 fournit des informations sur la façon dont l'opération LN affecte l'état de signal des bits d'état BI1, BI0, DEB et DM.

Tableau 12-10 Logarithme naturel d'un nombre réel : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN	REAL	E, A, M, D, L	Nombre réel
	OUT	REAL	E, A, M, D, L	Logarithme naturel du nombre réel



L'opération LN est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le résultat de LN (MD0) est rangé dans le double mot de mémento MD10. Si MD0 < 0 ou si le résultat est hors de la plage autorisée pour un nombre réel ou si l'état de signal de l'entrée E0.0 est égal à 0, la sortie A 4.0 est mise à 1.

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	x	x	x	x	0	x	x	1

Figure 12-7 Logarithme naturel d'un nombre réel

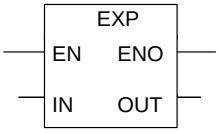
12.10 Valeur exponentielle d'un nombre réel

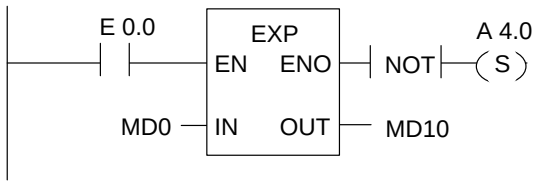
Description

L'opération « Valeur exponentielle d'un nombre réel » vous permet d'obtenir la valeur exponentielle de base e (= 2,71828...) d'un nombre réel.

Le paragraphe 12.6 fournit des informations sur la façon dont l'opération EXP affecte l'état de signal des bits d'état BI1, BI0, DEB et DM du mot d'état.

Tableau 12-11 Valeur exponentielle d'un nombre réel : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN	REAL	E, A, M, D, L	Nombre réel
	OUT	REAL	E, A, M, D, L	Valeur exponentielle du nombre réel



L'opération EXP est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le résultat de EXP (MD0) est rangé dans le double mot de mémento MD10. Si le résultat est hors de la plage autorisée pour un nombre réel ou si l'état de signal de l'entrée E0.0 est égal à 0, la sortie A 4.0 est mise à 1.

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	x	x	x	x	0	x	x	1

Figure 12-8 Valeur exponentielle d'un nombre réel

12.11 Fonctions trigonométriques d'angles sous forme de nombres réels

Description

Les opérations suivantes vous permettent d'affecter des fonctions trigonométriques d'angles représentés sous forme de nombres à virgule flottante IEEE de 32 bits :

Opération	Signification
SIN	Sinus d'un nombre réel d'un angle exprimé en radians
ASIN	Arc sinus d'un nombre réel. Le résultat de l'opération est un angle exprimé en radians. La valeur de cet angle est comprise dans la plage suivante : $\pi / 2 \leq$ arc sinus $\leq + \pi / 2$, avec $\pi = 3,14...$
COS	Cosinus d'un nombre réel d'un angle exprimé en radians
ACOS	Arc cosinus d'un nombre réel. Le résultat de l'opération est un angle exprimé en radians. La valeur de cet angle est comprise dans la plage suivante : $0 \leq$ arc cosinus $\leq + \pi$, avec $\pi = 3,14...$
TAN	Tangente d'un nombre réel d'un angle exprimé en radians
ATAN	Arc tangente d'un nombre réel. Le résultat de l'opération est un angle exprimé en radians. La valeur de cet angle est comprise dans la plage suivante : $\pi / 2 \leq$ arc tangente $\leq + \pi / 2$, avec $\pi = 3,14...$

Le paragraphe 12.6 fournit des informations sur la façon dont les opérations SIN, ASIN, COS, ACOS, TAN et ATAN affectent l'état de signal des bits d'état BI1, BI0, DEB et DM du mot d'état.

Paramètres

Les tableaux 12-12 à 12-17 donnent une représentation des pavés SIN, ASIN, COS, ACOS, TAN et ATAN et une description des paramètres.

Tableau 12-12 Sinus d'un nombre réel : représentation et paramètres

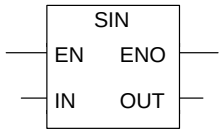
Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN	REAL	E, A, M, D, L	Nombre réel
	OUT	REAL	E, A, M, D, L	Sinus du nombre réel

Tableau 12-13 Arc sinus d'un nombre réel : représentation et paramètres

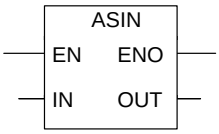
Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN	REAL	E, A, M, D, L	Nombre réel
	OUT	REAL	E, A, M, D, L	Arc sinus du nombre réel

Tableau 12-14 Cosinus d'un nombre réel : représentation et paramètres

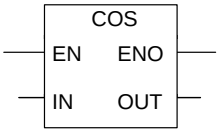
Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN	REAL	E, A, M, D, L	Nombre réel
	OUT	REAL	E, A, M, D, L	Cosinus du nombre réel

Tableau 12-15 Arc cosinus d'un nombre réel : représentation et paramètres

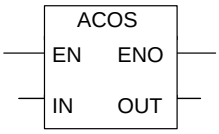
Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN	REAL	E, A, M, D, L	Nombre réel
	OUT	REAL	E, A, M, D, L	Arc cosinus du nombre réel

Tableau 12-16 Tangente : représentation et paramètres

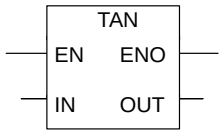
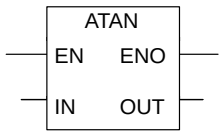
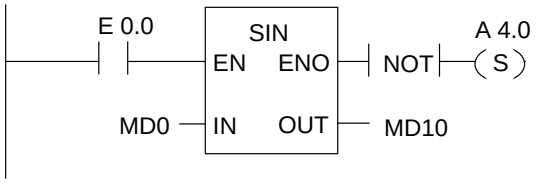
Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN	REAL	E, A, M, D, L	Nombre réel
	OUT	REAL	E, A, M, D, L	Tangente du nombre réel

Tableau 12-17 Arc tangente : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN	REAL	E, A, M, D, L	Nombre réel
	OUT	REAL	E, A, M, D, L	Arc tangente du nombre réel



L'opération SIN est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le résultat de SIN (MD0) est rangé dans le double mot de mémento MD10. Si le résultat est hors de la plage autorisée pour un nombre réel ou si l'état de signal de l'entrée E0.0 est égal à 0, la sortie A 4.0 est mise à 1.

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	x	x	x	x	0	x	x	1

Figure 12-9 Sinus d'un nombre réel

Opérations de comparaison

13

Contenu de ce chapitre

Paragraphe	Thème	Page
13.1	Comparer entiers de 16 bits	13-2
13.2	Comparer entiers de 32 bits	13-3
13.3	Comparer nombres réels	13-5

13.1 Comparer entiers de 16 bits

Description

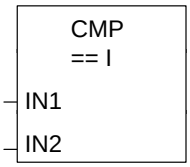
L'opération « Comparer entiers de 16 bits » que vous pouvez utiliser comme un contact normal compare les entrées IN1 et IN2 selon le type de comparaison que vous sélectionnez dans le champ de dialogue. Le tableau 13-1 présente les comparaisons autorisées.

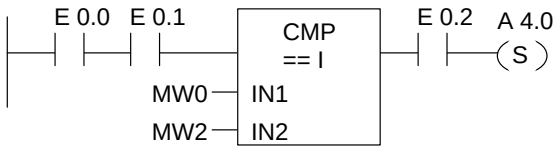
Si la comparaison est vraie, son résultat logique (RLG) est 1 ; dans le cas contraire, le RLG est 0. Il n'y a pas de négation de la sortie de comparaison, car cela peut être obtenu à l'aide de l'opération de comparaison inverse.

Tableau 13-1 Types de comparaison pour les nombres entiers de 16 bits

Type de comparaison	Symbole figurant dans le titre du pavé
IN1 est égal à IN2.	==
IN1 est différent de IN2.	<>
IN1 est supérieur à IN2.	>
IN1 est inférieur à IN2.	<
IN1 est supérieur ou égal à IN2.	>=
IN1 est inférieur ou égal à IN2.	<=

Tableau 13-2 Comparer entiers de 16 bits : représentation et paramètres (exemple : égal à)

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	IN1	INT	E, A, M, D, L	Premier terme de la comparaison
	IN2	INT	E, A, M, D, L	Second terme de la comparaison



La sortie A 4.0 est mise à 1 si les conditions suivantes sont satisfaites :

- L'état de signal est 1 aux entrées E 0.0 ET E 0.1
- ET MW0 = MW2
- ET l'état de signal est 1 à l'entrée E 0.2.

Description des bits du mot d'état

Comparaison vraie :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	x	x	0	–	0	1	x	1

Figure 13-1 Comparer entiers de 16 bits

13.2 Comparer entiers de 32 bits

Description

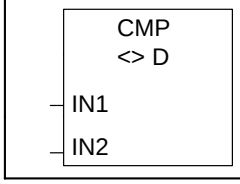
L'opération « Comparer entiers de 32 bits » que vous pouvez utiliser comme un contact normal compare les entrées IN1 et IN2 selon le type de comparaison que vous sélectionnez dans le champ de dialogue. Le tableau 13-3 présente les comparaisons autorisées.

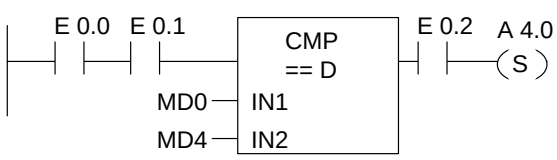
Si la comparaison est vraie, son résultat logique (RLG) est 1 ; dans le cas contraire, le RLG est 0. Il n'y a pas de négation de la sortie de comparaison, car cela peut être obtenu à l'aide de l'opération de comparaison inverse.

Tableau 13-3 Types de comparaison pour les nombres entiers de 32 bits

Type de comparaison	Symbole figurant dans le titre du pavé
IN1 est égal à IN2.	==
IN1 est différent de IN2.	<>
IN1 est supérieur à IN2.	>
IN1 est inférieur à IN2.	<
IN1 est supérieur ou égal à IN2.	>=
IN1 est inférieur ou égal à IN2.	<=

Tableau 13-4 Comparer entiers de 32 bits : représentation et paramètres (exemple : différent de)

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	IN1	DINT	E, A, M, D, L	Premier terme de la comparaison
	IN2	DINT	E, A, M, D, L	Second terme de la comparaison



La sortie A 4.0 est mise à 1 si les conditions suivantes sont satisfaites :

- L'état de signal est 1 aux entrées E 0.0 ET E 0.1
- ET MD0 est égal à MD4
- ET l'état de signal est 1 à l'entrée E 0.2.

Description des bits du mot d'état

Comparaison vraie :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	–	x	x	0	–	0	x	x	1

Figure 13-2 Comparer entiers de 32 bits

13.3 Comparer nombres réels

Description

L'opération « Comparer nombres réels » que vous pouvez utiliser comme un contact normal compare les entrées IN1 et IN2 selon le type de comparaison que vous sélectionnez dans le champ de dialogue. Le tableau 13-5 présente les comparaisons autorisées.

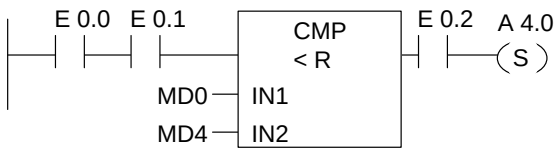
Si la comparaison est vraie, son résultat logique (RLG) est 1 ; dans le cas contraire, le RLG est 0. Il n'y a pas de négation de la sortie de comparaison, car cela peut être obtenu à l'aide de l'opération de comparaison inverse.

Tableau 13-5 Types de comparaison pour les nombres réels

Type de comparaison	Symbole figurant dans le titre du pavé
IN1 est égal à IN2.	==
IN1 est différent de IN2.	<>
IN1 est supérieur à IN2.	>
IN1 est inférieur à IN2.	<
IN1 est supérieur ou égal à IN2.	>=
IN1 est inférieur ou égal à IN2.	<=

Tableau 13-6 Comparer nombres réels : représentation et paramètres (exemple : inférieur à)

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
CMP < R IN1 IN2	IN1	REAL	E, A, M, D, L	Premier terme de la comparaison
	IN2	REAL	E, A, M, D, L	Second terme de la comparaison



La sortie A 4.0 est mise à 1 si les conditions suivantes sont satisfaites :

- L'état de signal est 1 aux entrées E 0.0 ET E 0.1
- ET MD0 est inférieur à MD4
- ET l'état de signal est 1 à l'entrée E 0.2.

Description des bits du mot d'état

Comparaison vraie :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	–	x	x	x	x	0	x	x	1

Figure 13-3 Comparer nombres réels

Opérations de transfert et de conversion 14

Contenu de ce chapitre

Paragraphe	Thème	Page
14.1	Affecter valeur	14-2
14.2	Convertir nombre DCB en entier de 16 bits	14-4
14.3	Convertir entier de 16 bits en nombre DCB	14-5
14.4	Convertir entier de 16 bits en entier de 32 bits	14-6
14.5	Convertir nombre DCB en entier de 32 bits	14-7
14.6	Convertir entier de 32 bits en nombre DCB	14-8
14.7	Convertir entier de 32 bits en nombre réel	14-9
14.8	Complément à 1 d’entier de 16 bits	14-10
14.9	Complément à 1 d’entier de 32 bits	14-11
14.10	Complément à 2 d’entier de 16 bits	14-12
14.11	Complément à 2 d’entier de 32 bits	14-13
14.12	Inverser le signe d’un nombre réel	14-14
14.13	Arrondir à entier de 32 bits	14-15
14.14	Tronquer à la partie entière (32 bits)	14-16
14.15	Convertir nombre réel en entier supérieur le plus proche	14-17
14.16	Convertir nombre réel en entier inférieur le plus proche	14-18

14.1 Affecter valeur

Description

L'opération « Affecter valeur » permet de prédéfinir une variable avec une valeur donnée.

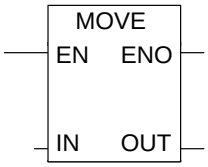
La valeur indiquée dans l'entrée IN est copiée à l'adresse précisée dans la sortie OUT. L'état de signal de ENO est identique à celui de EN.

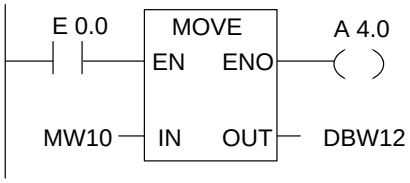
Avec le pavé « MOVE », l'opération « Affecter valeur » permet de copier tous les types de données de 8, 16 ou 32 bits. Pour copier des types de données utilisateur tels que des tableaux ou des structures, vous devez faire appel à la fonction système intégrée « Copier variable » (voir manuel de programmation **/234/**).

L'opération « Affecter valeur » est affectée par le relais de masquage (MCR). Pour plus d'informations sur le fonctionnement du MCR, voir le paragraphe 20.5.

Lorsque vous placez le pavé « Affecter valeur », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 14-1 Affecter valeur : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN	Tous types de données de 8, 16 ou 32 bits	E, A, M, D, L	Valeur source
	OUT	Tous types de données de 8, 16 ou 32 bits	E, A, M, D, L	Adresse de destination



L'opération est exécutée si l'état de signal de l'entrée E 0.0 est à 1. Le contenu du mot de mémento MW10 est copié dans le mot de données 12 du bloc de données ouvert.

A 4.0 est mis à 1 si l'opération est exécutée.

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	1	–	–	–	–	0	1	1	1

Figure 14-1 Affecter valeur

Affecter une valeur donnée à une variable

Pour obtenir des informations sur les fonctions système utilisées comme opérations de transfert capables d'affecter une valeur donnée à une variable ou de copier des variables de divers types, consultez le manuel de programmation **/234/**.

14.2 Convertir nombre DCB en entier de 16 bits

Description

L'opération « Convertir nombre DCB en entier de 16 bits » lit le contenu du paramètre d'entrée IN comme nombre décimal codé binaire à trois chiffres (DCB, ± 999), le convertit en un nombre entier de 16 bits et range le résultat dans le paramètre de sortie OUT.

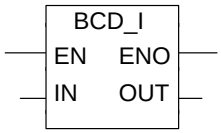
ENO et EN ont toujours un état de signal identique.

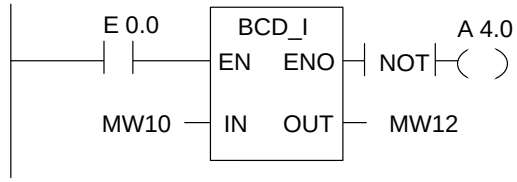
Si l'un des chiffres du nombre DCB se trouve dans la plage non autorisée entre 10 et 15, une erreur DCB se produit durant la tentative de conversion.

- La CPU passe à l'arrêt (STOP). La mémoire tampon de diagnostic indique une « Erreur de conversion DCB » de numéro d'événement 2521.
- L'OB121, s'il a été programmé, est appelé. Pour plus d'informations sur la programmation de l'OB121, voir le manuel de programmation **/234/**.

Lorsque vous placez le pavé « Convertir nombre DCB en entier de 16 bits », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 14-2 Convertir nombre DCB en entier de 16 bits : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN	WORD	E, A, M, D, L	Nombre en format DCB
	OUT	INT	E, A, M, D, L	Valeur entière de 16 bits du nombre DCB



La conversion est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le contenu du mot de mémento MW10 est lu comme nombre DCB à trois chiffres et converti en nombre entier de 16 bits. Le résultat est rangé dans le mot de mémento MW12. La sortie A 4.0 est mise à 1 si la conversion n'est pas exécutée (ENO = EN).

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	1	–	–	–	–	0	1	1	1

Figure 14-2 Convertir nombre DCB en entier de 16 bits

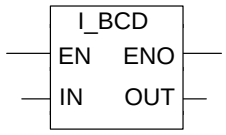
14.3 Convertir entier de 16 bits en nombre DCB

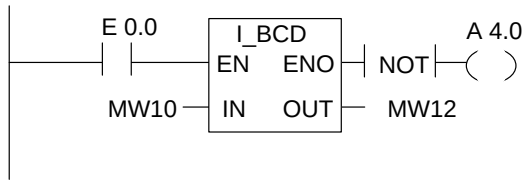
Description

L'opération « Convertir entier de 16 bits en nombre DCB » lit le contenu du paramètre d'entrée IN comme valeur entière de 16 bits, le convertit en un nombre décimal codé binaire à trois chiffres (DCB, ± 999) et range le résultat dans le paramètre de sortie OUT. En cas de débordement, ENO est mis à 0.

Lorsque vous placez le pavé « Convertir entier de 16 bits en nombre DCB », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 14-3 Convertir entier de 16 bits en nombre DCB : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN	INT	E, A, M, D, L	Nombre entier de 16 bits
	OUT	WORD	E, A, M, D, L	Résultat en format DCB



La conversion est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le contenu du mot de mémento MW10 est lu comme nombre entier de 16 bits et converti en nombre DCB à trois chiffres. Le résultat est rangé dans le mot de mémento MW12. En cas de débordement, la sortie A 4.0 est mise à 1. Si l'état de signal à l'entrée EN est égal à 0 (c'est-à-dire si la conversion n'est pas exécutée), la sortie A 4.0 est aussi mise à 1.

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	–	–	x	x	0	x	x	1

Figure 14-3 Convertir entier de 16 bits en nombre DCB

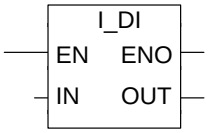
14.4 Convertir entier de 16 bits en entier de 32 bits

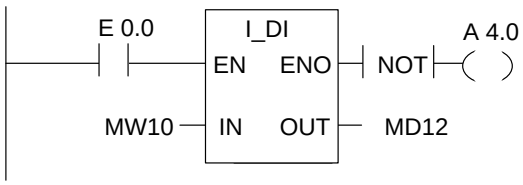
Description

L'opération « Convertir entier de 16 bits en entier de 32 bits » lit le contenu du paramètre d'entrée IN comme valeur entière de 16 bits, le convertit en un nombre entier de 32 bits et range le résultat dans le paramètre de sortie OUT. ENO et EN ont toujours un état de signal identique.

Lorsque vous placez le pavé « Convertir entier de 16 bits en entier de 32 bits », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 14-4 Convertir entier de 16 bits en entier de 32 bits : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN	INT	E, A, M, D, L	Valeur à convertir
	OUT	DINT	E, A, M, D, L	Résultat



La conversion est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le contenu du mot de mémento MW10 est lu comme nombre entier de 16 bits et converti en nombre entier de 32 bits. Le résultat est rangé dans le double mot de mémento MD12. La sortie A 4.0 est mise à 1 si la conversion n'est pas exécutée (ENO = EN).

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	1	–	–	–	–	0	1	1	1

Figure 14-4 Convertir entier de 16 bits en entier de 32 bits

14.5 Convertir nombre DCB en entier de 32 bits

Description

L'opération « Convertir nombre DCB en entier de 32 bits » lit le contenu du paramètre d'entrée IN comme nombre décimal codé binaire à sept chiffres (DCB, ± 9 999 999), le convertit en un nombre entier de 32 bits et range le résultat dans le paramètre de sortie OUT.

ENO et EN ont toujours un état de signal identique.

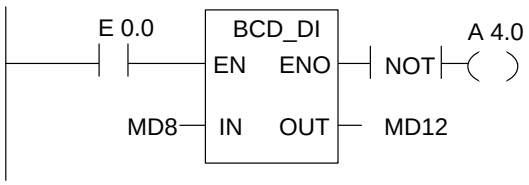
Si l'un des chiffres du nombre DCB se trouve dans la plage non autorisée entre 10 et 15, une erreur DCB se produit durant la tentative de conversion.

- La CPU passe à l'arrêt (STOP). La mémoire tampon de diagnostic indique une « Erreur de conversion DCB » de numéro d'événement 2521.
- L'OB121, s'il a été programmé, est appelé. Pour plus d'informations sur la programmation de l'OB121, voir le manuel de programmation **/234/**.

Lorsque vous placez le pavé « Convertir nombre DCB en entier de 32 bits », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 14-5 Convertir nombre DCB en entier de 32 bits : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
BCD_DI EN ENO IN OUT	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN	DWORD	E, A, M, D, L	Nombre en format DCB
	OUT	DINT	E, A, M, D, L	Valeur entière de 32 bits du nombre DCB



La conversion est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le contenu du double mot de mémento MD8 est lu comme nombre DCB à sept chiffres et converti en nombre entier de 32 bits. Le résultat est rangé dans le double mot de mémento MD12. La sortie A 4.0 est mise à 1 si la conversion n'est pas exécutée (ENO = EN).

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	1	–	–	–	–	0	1	1	1

Figure 14-5 Convertir nombre DCB en entier de 32 bits

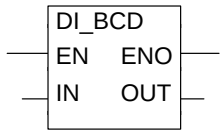
14.6 Convertir entier de 32 bits en nombre DCB

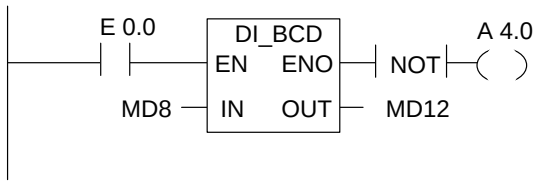
Description

L'opération « Convertir entier de 32 bits en nombre DCB » lit le contenu du paramètre d'entrée IN comme valeur entière de 32 bits, le convertit en un nombre décimal codé binaire à sept chiffres (DCB, ± 9 999 999) et range le résultat dans le paramètre de sortie OUT. En cas de débordement, ENO est mis à 0.

Lorsque vous placez le pavé « Convertir entier de 32 bits en nombre DCB », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 14-6 Convertir entier de 32 bits en nombre DCB : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN	DINT	E, A, M, D, L	Nombre entier de 32 bits
	OUT	DWORD	E, A, M, D, L	Résultat en format DCB



La conversion est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le contenu du double mot de mémento MD8 est lu comme nombre entier de 32 bits et converti en nombre DCB à sept chiffres. Le résultat est rangé dans le double mot de mémento MD12. En cas de débordement, la sortie A 4.0 est mise à 1. Si l'état de signal à l'entrée EN est égal à 0 (c'est-à-dire si la conversion n'est pas exécutée), la sortie A 4.0 est aussi mise à 1.

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	–	–	x	x	0	x	x	1

Figure 14-6 Convertir entier de 32 bits en nombre DCB

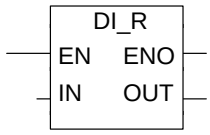
14.7 Convertir entier de 32 bits en nombre réel

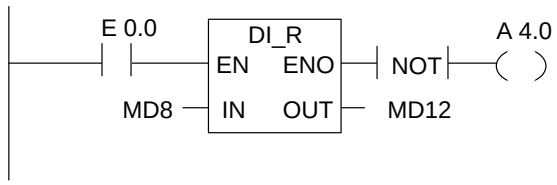
Description

L'opération « Convertir entier de 32 bits en nombre réel » lit le contenu du paramètre d'entrée IN comme valeur entière de 32 bits, le convertit en un nombre réel et range le résultat dans le paramètre de sortie OUT. ENO et EN ont toujours un état de signal identique.

Lorsque vous placez le pavé « Convertir entier de 32 bits en nombre réel », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 14-7 Convertir entier de 32 bits en nombre réel : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN	DINT	E, A, M, D, L	Valeur à convertir
	OUT	REAL	E, A, M, D, L	Résultat



La conversion est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le contenu du double mot de mémento MD8 est lu comme nombre entier de 32 bits et converti en nombre réel. Le résultat est rangé dans le double mot de mémento MD12. La sortie A 4.0 est mise à 1 si la conversion n'est pas exécutée (ENO = EN).

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	1	–	–	–	–	0	1	1	1

Figure 14-7 Convertir entier de 32 bits en nombre réel

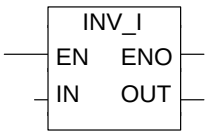
14.8 Complément à 1 d'entier de 16 bits

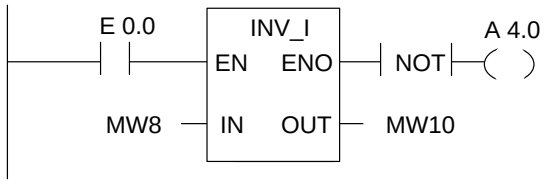
Description

L'opération « Complément à 1 d'entier de 16 bits » lit le contenu du paramètre d'entrée IN et exécute l'opération de combinaison « OU exclusif mot » (voir paragraphe 15.6) avec le masque hexadécimal $FFFF_H$ afin d'inverser la valeur de chaque bit. Le résultat est rangé dans le paramètre de sortie OUT. ENO et EN ont toujours un état de signal identique.

Lorsque vous placez le pavé « Complément à 1 d'entier de 16 bits », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 14-8 Complément à 1 d'entier de 16 bits : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN	INT	E, A, M, D, L	Valeur d'entrée
	OUT	INT	E, A, M, D, L	Complément à 1 de nombre entier de 16 bits



La conversion est exécutée si l'état de signal est 1 à l'entrée E 0.0. Chaque bit de MW8 est inversé :

MW8 = 01000001 10000001 →
MW10 = 10111110 01111110

La sortie A 4.0 est mise à 1 si la conversion n'est pas exécutée (ENO = EN).

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	1	–	–	–	–	0	1	1	1

Figure 14-8 Complément à 1 d'entier de 16 bits

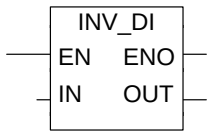
14.9 Complément à 1 d'entier de 32 bits

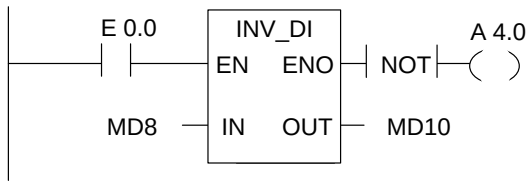
Description

L'opération « Complément à 1 d'entier de 32 bits » lit le contenu du paramètre d'entrée IN et exécute l'opération de combinaison « OU exclusif double mot » (voir paragraphe 15.6) avec le masque hexadécimal FFFF $FFFF_H$ afin d'inverser la valeur de chaque bit. Le résultat est rangé dans le paramètre de sortie OUT. ENO et EN ont toujours un état de signal identique.

Lorsque vous placez le pavé « Complément à 1 d'entier de 32 bits », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 14-9 Complément à 1 d'entier de 32 bits : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN	DINT	E, A, M, D, L	Valeur d'entrée
	OUT	DINT	E, A, M, D, L	Complément à 1 de nombre entier de 32 bits



La conversion est exécutée si l'état de signal est 1 à l'entrée E 0.0. Chaque bit du double mot de mémento MD8 est inversé :

MD8 = F0FF FFF0 → MD10 = 0F00 000F

La sortie A 4.0 est mise à 1 si la conversion n'est pas exécutée (ENO = EN).

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	1	–	–	–	–	0	1	1	1

Figure 14-9 Complément à 1 d'entier de 32 bits

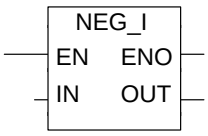
14.10 Complément à 2 d'entier de 16 bits

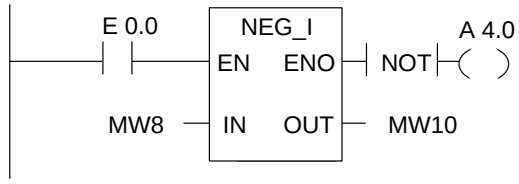
Description

L'opération « Complément à 2 d'entier de 16 bits » lit le contenu du paramètre d'entrée IN et en change le signe (par exemple, valeur positive en valeur négative). Le résultat est rangé dans le paramètre de sortie OUT. ENO et EN ont toujours un état de signal identique. Exception : si l'état de signal de EN est égal à 1 et qu'il y a débordement, alors l'état de signal de ENO est égal à 0.

Lorsque vous placez le pavé « Complément à 2 d'entier de 16 bits », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 14-10 Complément à 2 d'entier de 16 bits : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN	INT	E, A, M, D, L	Valeur d'entrée
	OUT	INT	E, A, M, D, L	Complément à 2 de nombre entier de 16 bits



La conversion est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le contenu du mot de mémento MW8 est rangé dans le paramètre de sortie OUT, dans le mot de mémento MW10, avec le signe opposé. Exemple :

MW8 = +10 → MW10 = – 10

Si l'état de signal de EN est égal à 1 et qu'il y a débordement, alors l'état de signal de ENO est égal à 0 et celui de la sortie A 4.0 est égal à 1. La sortie A 4.0 est mise à 1 si la conversion n'est pas exécutée (ENO = EN).

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	x	x	x	x	0	x	x	1

Figure 14-10 Complément à 2 d'entier de 16 bits

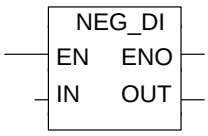
14.11 Complément à 2 d'entier de 32 bits

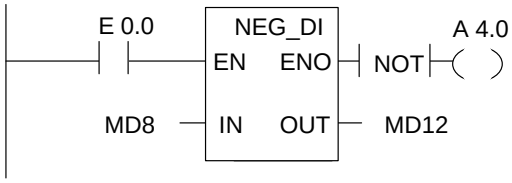
Description

L'opération « Complément à 2 d'entier de 32 bits » lit le contenu du paramètre d'entrée IN et en change le signe (par exemple, valeur positive en valeur négative). Le résultat est rangé dans le paramètre de sortie OUT. ENO et EN ont toujours un état de signal identique. Exception : si l'état de signal de EN est égal à 1 et qu'il y a débordement, alors l'état de signal de ENO est égal à 0.

Lorsque vous placez le pavé « Complément à 2 d'entier de 32 bits », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 14-11 Complément à 2 d'entier de 32 bits : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN	DINT	E, A, M, D, L	Valeur d'entrée
	OUT	DINT	E, A, M, D, L	Complément à 2 d'entier de 32 bits



La conversion est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le contenu du double mot de mémento MD8 est rangé dans le paramètre de sortie OUT, dans le double mot de mémento MD12, avec le signe opposé. Exemple :

MD8 = +60.000 → MD12 = – 60.000

Si l'état de signal de EN est égal à 1 et qu'il y a débordement, alors l'état de signal de ENO est égal à 0 et celui de la sortie A 4.0 est égal à 1. La sortie A 4.0 est mise à 1 si la conversion n'est pas exécutée (ENO = EN).

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	x	x	x	x	0	x	x	1

Figure 14-11 Complément à 2 d'entier de 32 bits

14.12 Inverser le signe d'un nombre réel

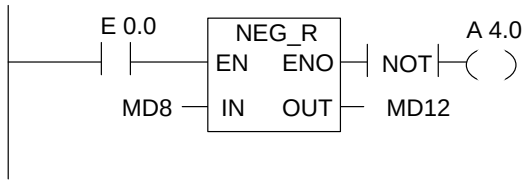
Description

L'opération « Inverser le signe d'un nombre réel » lit le contenu du paramètre d'entrée IN et en change le bit de signe (par exemple, de 0 pour une valeur positive en 1 pour une valeur négative). Les bits de l'exposant et de la mantisse restent les mêmes. Le résultat est rangé dans le paramètre de sortie OUT. ENO et EN ont toujours un état de signal identique.

Lorsque vous placez le pavé « Inverser le signe d'un nombre réel », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 14-12 Inverser le signe d'un nombre réel : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
NEG_R EN ENO IN OUT	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN	REAL	E, A, M, D, L	Valeur d'entrée
	OUT	REAL	E, A, M, D, L	Le résultat est la valeur négative de la valeur d'entrée.



La conversion est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le contenu du double mot de mémento MD8 est rangé dans le paramètre de sortie OUT, dans le double mot de mémento MD12, avec le signe opposé. Exemple :

$MD8 = + 6{,}234 \times 10^{-3} \rightarrow MD12 = - 6{,}234 \times 10^{-3}$

La sortie A 4.0 est mise à 1 si la conversion n'est pas exécutée (ENO = EN).

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	–	–	–	–	0	x	x	1

Figure 14-12 Inverser le signe d'un nombre réel

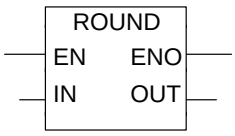
14.13 Arrondir à entier de 32 bits

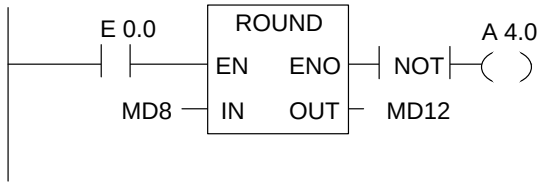
Description

L'opération « Arrondir à entier de 32 bits » lit le contenu du paramètre d'entrée IN comme nombre réel et le convertit en nombre entier de 32 bits. Le résultat, qui est le nombre entier le plus proche, est rangé dans le paramètre de sortie OUT. Si la partie fractionnaire est égale à ,5, le résultat fourni est le nombre pair. En cas de débordement, ENO est mis à 0.

Lorsque vous placez le pavé « Arrondir à entier de 32 bits », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 14-13 Arrondir à entier de 32 bits : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN	REAL	E, A, M, D, L	Valeur à arrondir
	OUT	DINT	E, A, M, D, L	IN arrondi au nombre entier le plus proche



La conversion est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le contenu du double mot de mémento MD8 est lu comme nombre réel et converti en nombre entier de 32 bits, avec arrondi au plus proche. Le résultat est rangé dans le double mot de mémento MD12. En cas de débordement, la sortie A 4.0 est mise à 1. Si l'état de signal de l'entrée EN est égal à 0 (c'est-à-dire si la conversion n'est pas exécutée), la sortie A 4.0 est aussi mise à 1.

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	–	–	x	x	0	x	x	1

Figure 14-13 Arrondir à entier de 32 bits

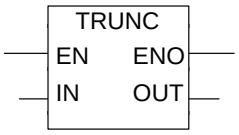
14.14 Tronquer à la partie entière (32 bits)

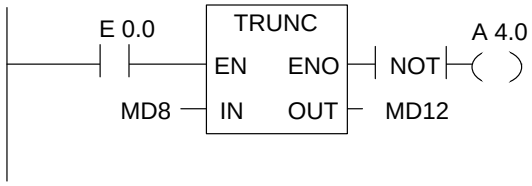
Description

L'opération « Tronquer à la partie entière (32 bits) » lit le contenu du paramètre d'entrée IN comme nombre réel et le convertit en nombre entier de 32 bits. Le résultat, qui est la partie entière du nombre réel spécifié, est rangé dans le paramètre de sortie OUT. En cas de débordement, ENO est mis à 0.

Lorsque vous placez le pavé « Tronquer à la partie entière (32 bits) », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 14-14 Tronquer à la partie entière (32 bits) : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN	REAL	E, A, M, D, L	Valeur à arrondir
	OUT	DINT	E, A, M, D, L	Partie entière de IN



La conversion est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le contenu du double mot de mémento MD8 est lu comme nombre réel et converti en nombre entier de 32 bits. Le résultat, qui est la composante entière, est rangé dans le double mot de mémento MD12. En cas de débordement, la sortie A 4.0 est mise à 1. Si l'état de signal de l'entrée EN est égal à 0 (c'est-à-dire si la conversion n'est pas exécutée), la sortie A 4.0 est aussi mise à 1.

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	–	–	x	x	0	x	x	1

Figure 14-14 Tronquer à la partie entière (32 bits)

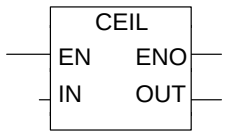
14.15 Convertir nombre réel en entier supérieur le plus proche

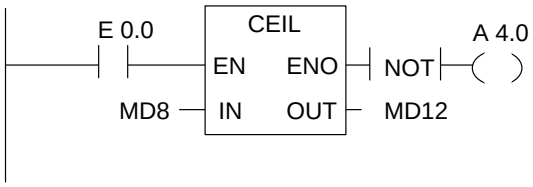
Description

L'opération « Convertir nombre réel en entier supérieur le plus proche » lit le contenu du paramètre d'entrée IN comme nombre réel et le convertit en un nombre entier de 32 bits. Le résultat, qui est la composante entière supérieure ou égale la plus proche du nombre réel indiqué, est rangé dans le paramètre de sortie OUT. En cas de débordement, ENO est mis à 0.

Lorsque vous placez le pavé « Convertir nombre réel en entier supérieur le plus proche », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 14-15 Convertir nombre réel en entier supérieur le plus proche : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN	REAL	E, A, M, D, L	Valeur à convertir
	OUT	DINT	E, A, M, D, L	Résultat



La conversion est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le contenu du double mot de mémento MD8 est lu comme nombre réel et converti en nombre entier de 32 bits selon le principe d'arrondi au nombre entier supérieur ou égal le plus proche. Le résultat est rangé dans le double mot de mémento MD12. En cas de débordement, la sortie A 4.0 est mise à 1. Si l'état de signal à l'entrée EN est égal à 0 (c'est-à-dire si la conversion n'est pas exécutée), la sortie A 4.0 est aussi mise à 1.

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	–	–	x	x	0	x	x	1

Figure 14-15 Convertir nombre réel en entier supérieur le plus proche

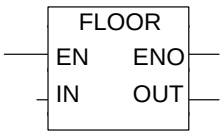
14.16 Convertir nombre réel en entier inférieur le plus proche

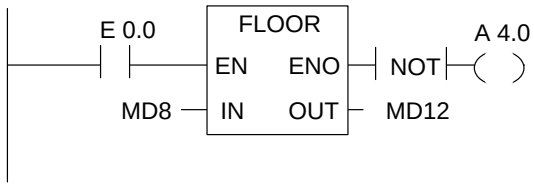
Description

L'opération « Convertir nombre réel en entier inférieur le plus proche » lit le contenu du paramètre d'entrée IN comme nombre réel et le convertit en un nombre entier de 32 bits. Le résultat, qui est la composante entière inférieure ou égale la plus proche du nombre réel indiqué, est rangé dans le paramètre de sortie OUT. En cas de débordement, ENO est mis à 0.

Lorsque vous placez le pavé « Convertir nombre réel en entier inférieur le plus proche », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 14-16 Convertir nombre réel en entier inférieur le plus proche : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN	REAL	E, A, M, D, L	Valeur à convertir
	OUT	DINT	E, A, M, D, L	Résultat



La conversion est exécutée si l'état de signal est 1 à l'entrée E 0.0. Le contenu du double mot de mémento MD8 est lu comme nombre réel et converti en nombre entier de 32 bits selon le principe d'arrondi au nombre entier inférieur (ou égal) le plus proche. Le résultat est rangé dans le double mot de mémento MD12. En cas de débordement, la sortie A 4.0 est mise à 1. Si l'état de signal à l'entrée EN est égal à 0 (c'est-à-dire si la conversion n'est pas exécutée), la sortie A 4.0 est aussi mise à 1.

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	–	–	x	x	0	x	x	1

Figure 14-16 Convertir nombre réel en entier inférieur le plus proche

Opérations combinatoires sur mots

15

Contenu de ce chapitre

Paragraphe	Thème	Page
15.1	Présentation	15-2
15.2	ET mot	15-3
15.3	ET double mot	15-4
15.4	OU mot	15-5
15.5	OU double mot	15-6
15.6	OU exclusif mot	15-7
15.7	OU exclusif double mot	15-8

15.1 Présentation

Opérations combinatoires sur mots

Les opérations combinatoires sur mots combinent deux mots (16 bits) ou deux doubles mots (32 bits), bit par bit, selon les combinaisons booléennes. Vous disposez des opérations combinatoires sur mots suivantes :

- « ET mot » combine deux mots bit par bit selon la table de vérité ET.
- « ET double mot » combine deux doubles mots bit par bit selon la table de vérité ET.
- « OU mot » combine deux mots bit par bit selon la table de vérité OU.
- « OU double mot » combine deux doubles mots bit par bit selon la table de vérité OU.
- « OU exclusif mot » combine deux mots bit par bit selon la table de vérité OU exclusif.
- « OU exclusif double mot » combine deux doubles mots bit par bit selon la table de vérité OU exclusif.

15.2 ET mot

Description

L'opération « ET mot » est activée si l'état de signal est 1 à l'entrée de validation EN. Cette opération combine, bit par bit selon la table de vérité ET, les deux mots indiqués dans les entrées IN1 et IN2. Ces valeurs sont interprétées comme profils binaires purs. Le résultat est rangé dans la sortie OUT. L'état de signal de ENO est identique à celui de EN.

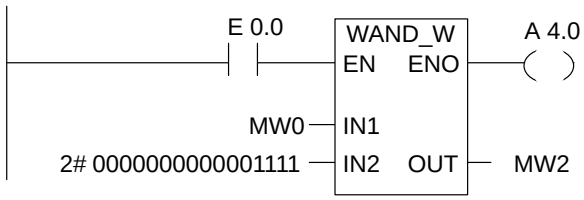
A la sortie OUT, la relation du résultat par rapport à 0 influence le bit indicateur BI1 du mot d'état de la manière suivante :

- Si, à la sortie OUT, le résultat est différent de 0, le bit indicateur BI1 du mot d'état est mis à 1.
- Si, à la sortie OUT, le résultat est égal à 0, le bit indicateur BI1 du mot d'état est égal à 0.

Lorsque vous placez des pavés d'opérations combinatoires sur mots, vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 15-1 ET mot : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
WAND_W EN ENO IN1 IN2 OUT	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN1	WORD	E, A, M, D, L	Première valeur de la combinaison
	IN2	WORD	E, A, M, D, L	Seconde valeur de la combinaison
	OUT	WORD	E, A, M, D, L	Résultat de la combinaison



L'opération « ET mot » est exécutée si l'état de signal est 1 à l'entrée E 0.0. Seuls les bits 0 à 3 sont significatifs ; les autres bits du mot de mémento MW0 sont masqués :

```
IN1   =   0101010101010101
IN2   =   0000000000001111
OUT   =   0000000000000101
```

La sortie A 4.0 est mise à 1 si l'opération est exécutée.

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	1	x	0	0	–	x	1	1	1

Figure 15-1 ET mot

15.3 ET double mot

Description

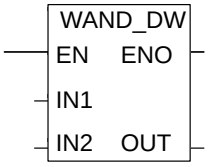
L'opération « ET double mot » est activée si l'état de signal est 1 à l'entrée de validation EN. Cette opération combine, bit par bit selon la table de vérité ET, les deux doubles mots indiqués dans les entrées IN1 et IN2. Ces valeurs sont interprétées comme profils binaires purs. Le résultat est rangé dans la sortie OUT. L'état de signal de ENO est identique à celui de EN.

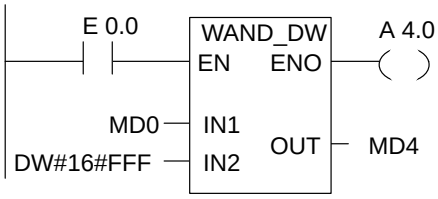
A la sortie OUT, la relation du résultat par rapport à 0 influence le bit indicateur BI1 du mot d'état de la manière suivante :

- Si, à la sortie OUT, le résultat est différent de 0, le bit indicateur BI1 du mot d'état est mis à 1.
- Si, à la sortie OUT, le résultat est égal à 0, le bit indicateur BI1 du mot d'état est égal à 0.

Lorsque vous placez des pavés d'opérations combinatoires sur mots, vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 15-2 ET double mot : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN1	DWORD	E, A, M, D, L	Première valeur de la combinaison
	IN2	DWORD	E, A, M, D, L	Seconde valeur de la combinaison
	OUT	DWORD	E, A, M, D, L	Résultat de la combinaison



L'opération « ET double mot » est exécutée si l'état de signal est 1 à l'entrée E 0.0. Seuls les bits 0 à 11 sont significatifs ; les autres bits du double mot de mémento MD0 sont masqués :

```
IN1  = 01010101010101010101010101010101
IN2  = 00000000000000000000111111111111
OUT  = 00000000000000000000010101010101
```

La sortie A 4.0 est mise à 1 si l'opération est exécutée.

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	1	x	0	0	–	x	1	1	1

Figure 15-2 ET double mot

15.4 OU mot

Description

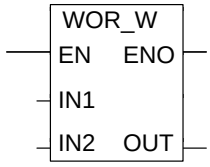
L'opération « OU mot » est activée si l'état de signal est 1 à l'entrée de validation EN. Cette opération combine, bit par bit selon la table de vérité OU, les deux mots indiqués dans les entrées IN1 et IN2. Ces valeurs sont interprétées comme profils binaires purs. Le résultat est rangé dans la sortie OUT. L'état de signal de ENO est identique à celui de EN.

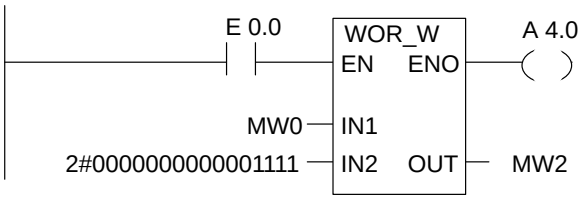
A la sortie OUT, la relation du résultat par rapport à 0 influence le bit indicateur BI1 du mot d'état de la manière suivante :

- Si, à la sortie OUT, le résultat est différent de 0, le bit indicateur BI1 du mot d'état est mis à 1.
- Si, à la sortie OUT, le résultat est égal à 0, le bit indicateur BI1 du mot d'état est égal à 0.

Lorsque vous placez des pavés d'opérations combinatoires sur mots, vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 15-3 OU mot : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN1	WORD	E, A, M, D, L	Première valeur de la combinaison
	IN2	WORD	E, A, M, D, L	Seconde valeur de la combinaison
	OUT	WORD	E, A, M, D, L	Résultat de la combinaison



L'opération « OU mot » est exécutée si l'état de signal est 1 à l'entrée E 0.0. Seuls les bits 0 à 3 sont mis à 1 ; les autres bits du mot de mémento MW0 restent inchangés :

```
IN1  = 0101010101010101
IN2  = 0000000000001111
OUT  = 0101010101011111
```

La sortie A 4.0 est mise à 1 si l'opération est exécutée.

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	1	x	0	0	–	x	1	1	1

Figure 15-3 OU mot

15.5 OU double mot

Description

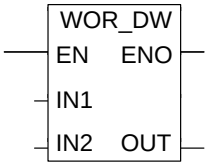
L'opération « OU double mot » est activée si l'état de signal est 1 à l'entrée de validation EN. Cette opération combine, bit par bit selon la table de vérité OU, les deux doubles mots indiqués dans les entrées IN1 et IN2. Ces valeurs sont interprétées comme profils binaires purs. Le résultat est rangé dans la sortie OUT. L'état de signal de ENO est identique à celui de EN.

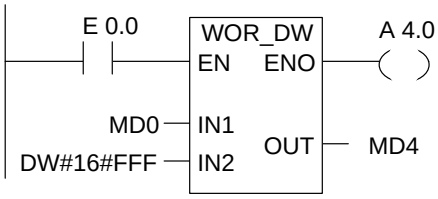
A la sortie OUT, la relation du résultat par rapport à 0 influence le bit indicateur BI1 du mot d'état de la manière suivante :

- Si, à la sortie OUT, le résultat est différent de 0, le bit indicateur BI1 du mot d'état est mis à 1.
- Si, à la sortie OUT, le résultat est égal à 0, le bit indicateur BI1 du mot d'état est égal à 0.

Lorsque vous placez des pavés d'opérations combinatoires sur mots, vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 15-4 OU double mot : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN1	DWORD	E, A, M, D, L	Première valeur de la combinaison
	IN2	DWORD	E, A, M, D, L	Seconde valeur de la combinaison
	OUT	DWORD	E, A, M, D, L	Résultat de la combinaison



L'opération « OU double mot » est exécutée si l'état de signal est 1 à l'entrée E 0.0. Les bits 0 à 11 sont mis à 1 ; les autres bits du double mot de mémento MD0 restent inchangés :

```
IN1  = 01010101010101011010101010101010
IN2  = 00000000000000000000111111111111
OUT  = 01010101010101011010111111111111
```

La sortie A 4.0 est mise à 1 si l'opération est exécutée.

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	1	x	0	0	–	x	1	1	1

Figure 15-4 OU double mot

15.6 OU exclusif mot

Description

L'opération « OU exclusif mot » est activée si l'état de signal est 1 à l'entrée de validation EN. Cette opération combine, bit par bit selon la table de vérité OU exclusif, les deux mots indiqués dans les entrées IN1 et IN2. Ces valeurs sont interprétées comme profils binaires purs. Le résultat est rangé dans la sortie OUT. L'état de signal de ENO est identique à celui de EN.

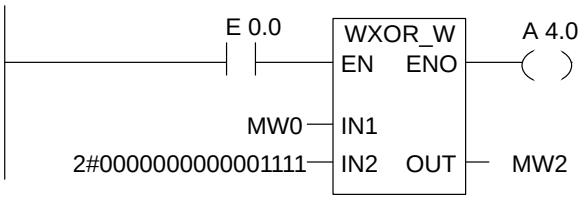
A la sortie OUT, la relation du résultat par rapport à 0 influence le bit indicateur BI1 du mot d'état de la manière suivante :

- Si, à la sortie OUT, le résultat est différent de 0, le bit indicateur BI1 du mot d'état est mis à 1.
- Si, à la sortie OUT, le résultat est égal à 0, le bit indicateur BI1 du mot d'état est égal à 0.

Lorsque vous placez des pavés d'opérations combinatoires sur mots, vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 15-5 OU exclusif mot : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
WXOR_W EN ENO IN1 IN2 OUT	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN1	WORD	E, A, M, D, L	Première valeur de la combinaison
	IN2	WORD	E, A, M, D, L	Seconde valeur de la combinaison
	OUT	WORD	E, A, M, D, L	Résultat de la combinaison



L'opération « OU exclusif mot » est exécutée si l'état de signal est 1 à l'entrée E 0.0.

```
IN1      = 0101010101010101
IN2      = 0000000000001111
OUT      = 0101010101011010
```

La sortie A 4.0 est mise à 1 si l'opération est exécutée.

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	1	x	0	0	–	x	1	1	1

Figure 15-5 OU exclusif mot

15.7 OU exclusif double mot

Description

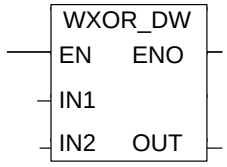
L'opération « OU exclusif double mot » est activée si l'état de signal est 1 à l'entrée de validation EN. Cette opération combine, bit par bit selon la table de vérité OU exclusif, les deux doubles mots indiqués dans les entrées IN1 et IN2. Ces valeurs sont interprétées comme profils binaires purs. Le résultat est rangé dans la sortie OUT. L'état de signal de ENO est identique à celui de EN.

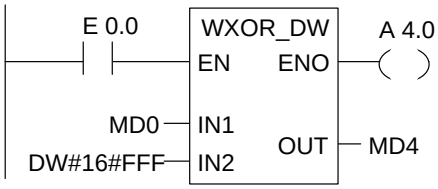
A la sortie OUT, la relation du résultat par rapport à 0 influence le bit indicateur BI1 du mot d'état de la manière suivante :

- Si, à la sortie OUT, le résultat est différent de 0, le bit indicateur BI1 du mot d'état est mis à 1.
- Si, à la sortie OUT, le résultat est égal à 0, le bit indicateur BI1 du mot d'état est égal à 0.

Lorsque vous placez des pavés d'opérations combinatoires sur mots, vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 15-6 OU exclusif double mot : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation
	IN1	DWORD	E, A, M, D, L	Première valeur de la combinaison
	IN2	DWORD	E, A, M, D, L	Seconde valeur de la combinaison
	OUT	DWORD	E, A, M, D, L	Résultat de la combinaison



L'opération « OU exclusif double mot » est exécutée si l'état de signal est 1 à l'entrée E 0.0.

```
IN1   = 0101010101010101 0101010101010101
IN2   = 0000000000000000 0000111111111111
OUT   = 0101010101010101 0101101010101010
```

La sortie A 4.0 est mise à 1 si l'opération est exécutée.

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	1	x	0	0	–	x	1	1	1

Figure 15-6 OU exclusif double mot

Opérations de décalage et de rotation

16

Contenu de ce chapitre

Paragraphe	Thème	Page
16.1	Opérations de décalage	16-2
16.2	Opérations de rotation	16-10

16.1 Opérations de décalage

Description

Les opérations de décalage permettent de décaler bit par bit le contenu de l'entrée IN vers la gauche ou vers la droite (voir paragraphe 6.3). Le décalage vers la gauche multiplie le contenu de l'entrée IN par des puissances de 2 ; le décalage vers la droite le divise par des puissances de 2. Si, par exemple, vous décalez de 3 bits vers la gauche l'équivalent binaire de la valeur décimale 3, vous obtenez l'équivalent binaire de la valeur décimale 24. Si vous décalez de 2 bits vers la droite l'équivalent binaire de la valeur décimale 16, vous obtenez l'équivalent binaire de la valeur décimale 4.

Le nombre de bits de décalage est précisé dans le paramètre d'entrée N. Les positions binaires libérées par l'opération de décalage sont soit remplies par des zéros, soit par l'état de signal du bit de signe (0 signifie positif et 1 négatif). L'état de signal du bit décalé en dernier est chargé dans le bit BI1 du mot d'état (voir paragraphe 6.3). Les bits BI0 et DEB du mot d'état sont remis à 0. Vous pouvez évaluer le bit BI1 à l'aide d'opérations de saut.

Vous disposez des opérations de décalage suivantes :

- Décalage vers gauche de mot, décalage vers gauche de double mot
- Décalage vers droite de mot, décalage vers droite de double mot
- Décalage vers droite d'entier de 16 bits, décalage vers droite d'entier de 32 bits

Décalage vers gauche de mot

L'opération « Décalage vers gauche de mot » est activée si l'état de signal est 1 à l'entrée de validation EN. Elle décale bit par bit vers la gauche les bits 0 à 15 de l'entrée IN.

Le nombre de bits de décalage est indiqué dans l'entrée N. Si N est supérieur à 16, la commande inscrit 0 dans la sortie OUT et met les bits BI0 et DEB du mot d'état à 0. Les positions binaires à droite sont complétées par des zéros. Le résultat du décalage est rangé dans la sortie OUT.

Lorsque N est différent de zéro, l'opération remet les bits BI0 et DEB du mot d'état à 0. Si elle est exécutée (EN = 1), ENO contient l'état de signal du bit décalé en dernier (identique à BI1 et RLG dans le mot d'état). Ainsi, les fonctions suivant cette opération et qui y sont connectées par ENO (cascade) ne sont pas exécutées si l'état de signal du dernier bit décalé était 0.

Lorsque vous placez le pavé « Décalage vers gauche de mot », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

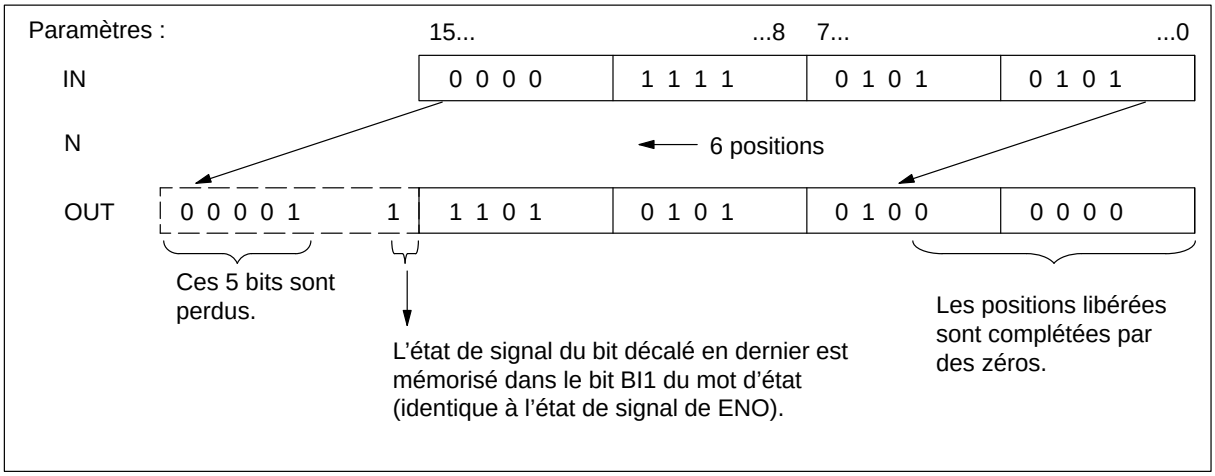
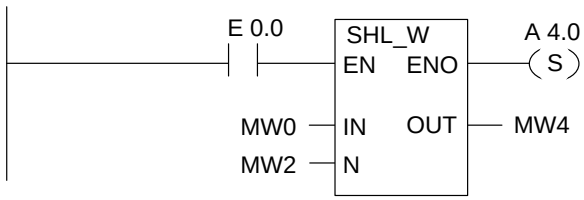


Figure 16-1 Décalage des bits de l'entrée IN de six positions vers la gauche

Tableau 16-1 Décalage vers gauche de mot : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
SHL_W EN ENO IN OUT N	EN	BOOL	E, A, M, L, D	Entrée de validation
	ENO	BOOL	E, A, M, L, D	Sortie de validation
	IN	WORD	E, A, M, L, D	Valeur à décaler
	N	WORD	E, A, M, L, D	Nombre de bits de décalage
	OUT	WORD	E, A, M, L, D	Résultat du décalage



L'opération est exécutée si l'état de signal est 1 à l'entrée E 0.0.

Le mot de mémento MW0 est décalé vers la gauche du nombre de bits précisé dans le mot de mémento MW2.

Le résultat est rangé dans le mot de mémento MW4. La sortie A 4.0 est mise à 1 si l'état de signal du bit décalé en dernier était 1.

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	x	x	x	–	x	x	x	1

Figure 16-2 Décalage vers gauche de mot

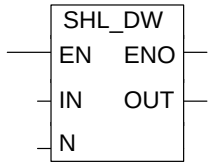
Décalage vers gauche de double mot

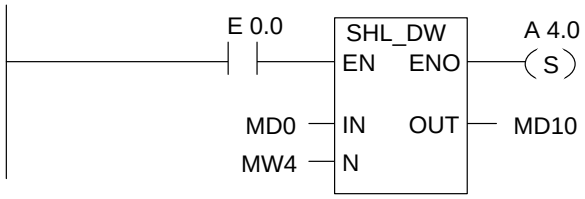
L'opération « Décalage vers gauche de double mot » est activée si l'état de signal est 1 à l'entrée de validation EN. Elle décale bit par bit vers la gauche les bits 0 à 31 de l'entrée IN. Le nombre de bits de décalage est indiqué dans l'entrée N. Si N est supérieur à 32, la commande inscrit 0 dans la sortie OUT et met les bits BI0 et DEB du mot d'état à 0. Les positions binaires à droite sont complétées par des zéros. Le résultat du décalage est rangé dans la sortie OUT.

Lorsque N est différent de zéro, l'opération remet toujours les bits BI0 et DEB du mot d'état à 0. Si elle est exécutée (EN = 1), ENO contient l'état de signal du bit décalé en dernier (identique à BI1 et RLG dans le mot d'état). Ainsi, les fonctions suivant cette opération et qui y sont connectées par ENO (cascade) ne sont pas exécutées si l'état de signal du dernier bit décalé était 0.

Lorsque vous placez le pavé « Décalage vers gauche de double mot », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 16-2 Décalage vers gauche de double mot : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, L, D	Entrée de validation
	ENO	BOOL	E, A, M, L, D	Sortie de validation
	IN	DWORD	E, A, M, L, D	Valeur à décaler
	N	WORD	E, A, M, L, D	Nombre de bits de décalage
	OUT	DWORD	E, A, M, L, D	Résultat du décalage



L'opération est exécutée si l'état de signal est 1 à l'entrée E 0.0.

Le double mot de mémento MD0 est décalé vers la gauche du nombre de bits précisé dans le mot de mémento MW4.

Le résultat est rangé dans le double mot de mémento MD10. La sortie A 4.0 est mise à 1 si l'état de signal du bit décalé en dernier était 1.

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	x	x	x	–	x	x	x	1

Figure 16-3 Décalage vers gauche de double mot

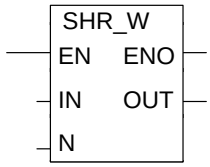
Décalage vers droite de mot

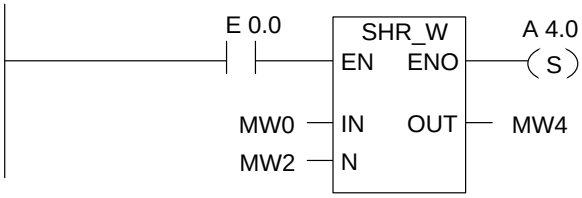
L'opération « Décalage vers droite de mot » est activée si l'état de signal est 1 à l'entrée de validation EN. Elle décale bit par bit vers la droite les bits 0 à 15 de l'entrée IN. Les bits 16 à 31 ne sont pas affectés. Le nombre de bits de décalage est indiqué dans l'entrée N. Si N est supérieur à 16, la commande inscrit 0 dans la sortie OUT et met les bits BI0 et DEB du mot d'état à 0. Les positions binaires à gauche sont complétées par des zéros. Le résultat du décalage est rangé dans la sortie OUT.

Lorsque N est différent de zéro, l'opération remet toujours les bits BI0 et DEB du mot d'état à 0. Si elle est exécutée (EN = 1), ENO contient l'état de signal du bit décalé en dernier (identique à BI1 et RLG dans le mot d'état). Ainsi, les fonctions suivant cette opération et qui y sont connectées par ENO (cascade) ne sont pas exécutées si l'état de signal du dernier bit décalé était 0.

Lorsque vous placez le pavé « Décalage vers droite de mot », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 16-3 Décalage vers droite de mot : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, L, D	Entrée de validation
	ENO	BOOL	E, A, M, L, D	Sortie de validation
	IN	WORD	E, A, M, L, D	Valeur à décaler
	N	WORD	E, A, M, L, D	Nombre de bits de décalage
	OUT	WORD	E, A, M, L, D	Résultat du décalage



L'opération est exécutée si l'état de signal est 1 à l'entrée E 0.0.

Le mot de mémento MW0 est décalé vers la droite du nombre de bits précisé dans le mot de mémento MW2.

Le résultat est rangé dans le mot de mémento MW4. La sortie A 4.0 est mise à 1 si l'état de signal du bit décalé en dernier était 1

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	x	x	x	–	x	x	x	1

Figure 16-4 Décalage vers droite de mot

Décalage vers droite de double mot

L'opération « Décalage vers droite de double mot » est activée si l'état de signal est 1 à l'entrée de validation EN. Elle décale bit par bit vers la droite les bits 0 à 31 de l'entrée IN. Le nombre de bits de décalage est indiqué dans l'entrée N. Si N est supérieur à 32, la commande inscrit 0 dans la sortie OUT et met les bits BI0 et DEB du mot d'état à 0. Les positions binaires à gauche sont complétées par des zéros. Le résultat du décalage est rangé dans la sortie OUT.

Lorsque N est différent de zéro, l'opération remet les bits BI0 et DEB du mot d'état à 0. Si elle est exécutée (EN = 1), ENO contient l'état de signal du bit décalé en dernier (identique à BI1 et RLG dans le mot d'état). Ainsi, les fonctions suivant cette opération et qui y sont connectées par ENO (cascade) ne sont pas exécutées si l'état de signal du dernier bit décalé était 0.

Lorsque vous placez le pavé « Décalage vers droite de double mot », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

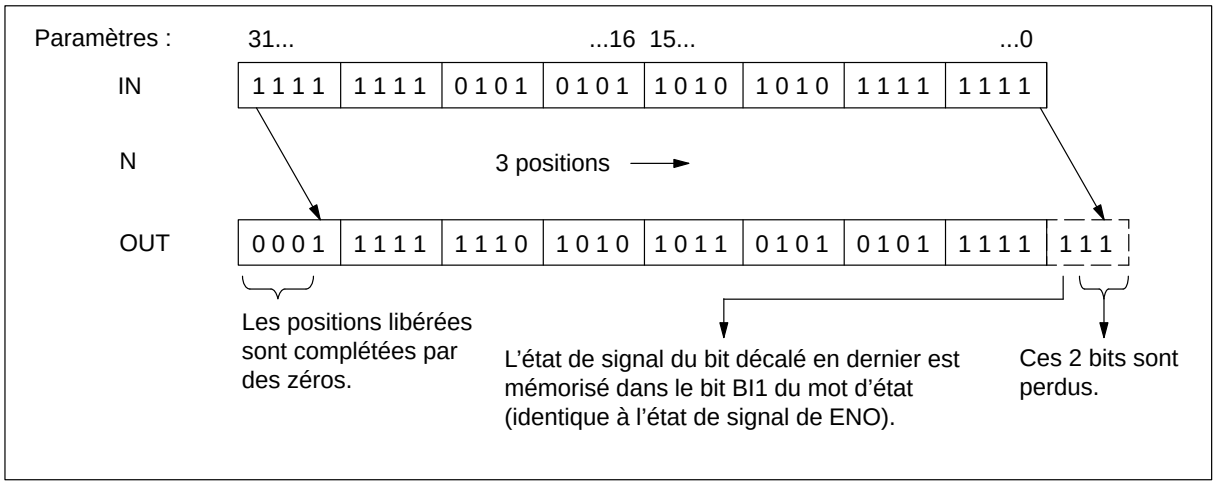
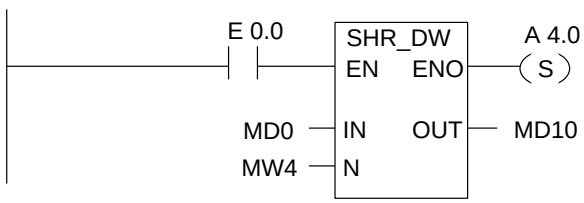

Paramètres :
31...
...16
15...
...0
IN
1 1 1 1
1 1 1 1
0 1 0 1
0 1 0 1
1 0 1 0
1 0 1 0
1 1 1 1
1 1 1 1
N
3 positions
OUT
0 0 0 1
1 1 1 1
1 1 1 0
1 0 1 0
1 0 1 1
0 1 0 1
0 1 0 1
1 1 1 1
1 1 1
Les positions libérées sont complétées par des zéros.
L'état de signal du bit décalé en dernier est mémorisé dans le bit BI1 du mot d'état (identique à l'état de signal de ENO).
Ces 2 bits sont perdus.

Figure 16-5 Décalage des bits de l'entrée IN de trois positions vers la droite

Tableau 16-4 Décalage vers droite de double mot : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
SHR_DW EN ENO IN OUT N	EN	BOOL	E, A, M, L, D	Entrée de validation
	ENO	BOOL	E, A, M, L, D	Sortie de validation
	IN	DWORD	E, A, M, L, D	Valeur à décaler
	N	WORD	E, A, M, L, D	Nombre de bits de décalage
	OUT	DWORD	E, A, M, L, D	Résultat du décalage



L'opération est exécutée si l'état de signal est 1 à l'entrée E 0.0.

Le double mot de mémento MD0 est décalé vers la droite du nombre de bits précisé dans le mot de mémento MW4.

Le résultat est rangé dans le double mot de mémento MD10. La sortie A 4.0 est mise à 1 si l'état de signal du bit décalé en dernier était 1.

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	x	x	x	–	x	x	x	1

Figure 16-6 Décalage vers droite de double mot

Décalage vers droite d'entier de 16 bits

L'opération « Décalage vers droite d'entier de 16 bits » est activée si l'état de signal est 1 à l'entrée de validation EN. Elle décale bit par bit vers la droite les bits 0 à 15 de l'entrée IN. Le nombre de bits de décalage est indiqué dans l'entrée N. Si N est supérieur à 16, la commande est exécutée comme si N était égal à 16. Les positions binaires à gauche sont complétées selon l'état de signal du bit 15 (bit de signe d'un nombre entier de 16 bits), c'est-à-dire par des zéros si le nombre est positif et par des uns si le nombre est négatif. Le résultat du décalage est rangé dans la sortie OUT.

Lorsque N est différent de zéro, l'opération remet les bits BI0 et DEB du mot d'état à 0. Si elle est exécutée (EN = 1), ENO contient l'état de signal du bit décalé en dernier (identique à BI1 et RLG dans le mot d'état). Ainsi, les fonctions suivant cette opération et qui y sont connectées par ENO (cascade) ne sont pas exécutées si l'état de signal du dernier bit décalé était 0.

Lorsque vous placez le pavé « Décalage vers droite d'entier de 16 bits », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

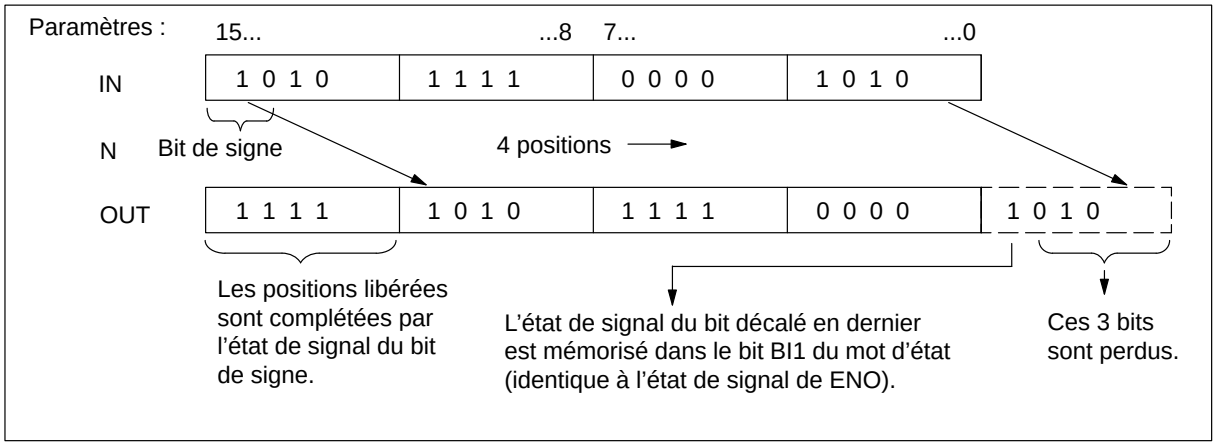
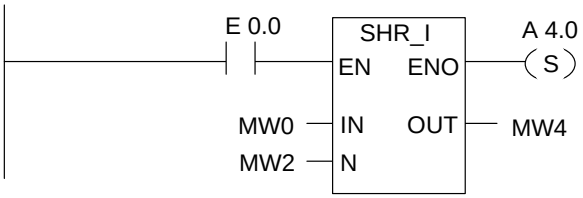

Paramètres :
15...
...8
7...
...0
IN
1 0 1 0
1 1 1 1
0 0 0 0
1 0 1 0
N
Bit de signe
4 positions
OUT
1 1 1 1
1 0 1 0
1 1 1 1
0 0 0 0
1 0 1 0
Les positions libérées sont complétées par l'état de signal du bit de signe.
L'état de signal du bit décalé en dernier est mémorisé dans le bit BI1 du mot d'état (identique à l'état de signal de ENO).
Ces 3 bits sont perdus.

Figure 16-7 Décalage des bits de l'entrée IN de quatre positions vers la droite avec le signe

Tableau 16-5 Décalage vers droite d'entier de 16 bits : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
SHR_I EN ENO IN OUT N	EN	BOOL	E, A, M, L, D	Entrée de validation
	ENO	BOOL	E, A, M, L, D	Sortie de validation
	IN	INT	E, A, M, L, D	Valeur à décaler
	N	WORD	E, A, M, L, D	Nombre de bits de décalage
	OUT	INT	E, A, M, L, D	Résultat du décalage



L'opération est exécutée si l'état de signal est 1 à l'entrée E 0.0.

Le mot de mémento MW0 est décalé vers la droite du nombre de bits précisé dans le mot de mémento MW2.

Le résultat est rangé dans le mot de mémento MW4. La sortie A 4.0 est mise à 1 si l'état de signal du bit décalé en dernier était 1.

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	x	x	x	–	x	x	x	1

Figure 16-8 Décalage vers droite d'entier de 16 bits

Décalage vers droite d'entier de 32 bits

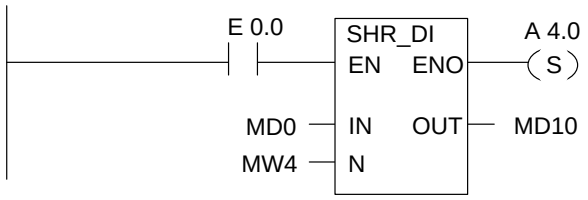
L'opération « Décalage vers droite d'entier de 32 bits » est activée si l'état de signal est 1 à l'entrée de validation EN. Elle décale bit par bit vers la droite le contenu de l'entrée IN. Le nombre de bits de décalage est indiqué dans l'entrée N. Si N est supérieur à 32, la commande est exécutée comme si N était égal à 32. Les positions binaires à gauche sont complétées selon l'état de signal du bit 31 (bit de signe d'un nombre entier de 32 bits), c'est-à-dire par des zéros si le nombre est positif et par des uns si le nombre est négatif. Le résultat du décalage est rangé dans la sortie OUT.

Lorsque N est différent de zéro, l'opération remet les bits BI0 et DEB du mot d'état à 0. Si elle est exécutée (EN = 1), ENO contient l'état de signal du bit décalé en dernier (identique à BI1 et RLG dans le mot d'état). Ainsi, les fonctions suivant cette opération et qui y sont connectées par ENO (cascade) ne sont pas exécutées si l'état de signal du dernier bit décalé était 0.

Lorsque vous placez le pavé « Décalage vers droite d'entier de 32 bits », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

Tableau 16-6 Décalage vers droite d'entier de 32 bits : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
SHR_DI EN ENO IN OUT N	EN	BOOL	E, A, M, L, D	Entrée de validation
	ENO	BOOL	E, A, M, L, D	Sortie de validation
	IN	DINT	E, A, M, L, D	Valeur à décaler
	N	WORD	E, A, M, L, D	Nombre de bits de décalage
	OUT	DINT	E, A, M, L, D	Résultat du décalage



L'opération est exécutée si l'état de signal est 1 à l'entrée E 0.0.

Le double mot de mémento MD0 est décalé vers la droite du nombre de bits précisé dans le mot de mémento MW4.

Le résultat est rangé dans le double mot de mémento MD10. La sortie A 4.0 est mise à 1 si l'état de signal du bit décalé en dernier était 1.

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	x	x	x	–	x	x	x	1

Figure 16-9 Décalage vers droite d'entier de 32 bits

16.2 Opérations de rotation

Description

Les opérations de rotation permettent d'effectuer la rotation bit par bit vers la droite ou vers la gauche du contenu entier de l'entrée IN. Les positions binaires libérées sont complétées par l'état de signal des bits qui ont été décalés hors de l'entrée IN.

Le nombre de bits de rotation est précisé dans le paramètre d'entrée N.

Selon l'opération, la rotation s'effectue via le bit BI1 du mot d'état (voir paragraphe 6.3). Le bit BI0 du mot d'état est remis à 0.

Vous disposez des opérations de rotation suivantes :

- Rotation vers gauche de double mot
- Rotation vers droite de double mot

Rotation vers gauche de double mot

L'opération « Rotation vers gauche de double mot » est activée si l'état de signal est 1 à l'entrée de validation EN. Elle déclenche la rotation bit par bit vers la gauche du contenu entier de l'entrée IN. Le nombre de bits de rotation est indiqué dans l'entrée N dont la valeur doit être comprise entre 0 et 31. Si N est supérieur à 32, le double mot effectue une rotation de ((N–1) modulo 32) + 1) positions vers la gauche. Les positions binaires à droite sont complétées par l'état de signal des bits objet de la rotation. Le résultat de la rotation est rangé dans la sortie OUT.

Lorsque N est différent de zéro, l'opération remet les bits BI0 et DEB du mot d'état à 0. Si elle est exécutée (EN = 1), ENO contient l'état de signal du dernier bit objet de la rotation (identique à BI1 et RLG dans le mot d'état). Ainsi, les fonctions suivant cette opération et qui y sont connectées par ENO (cascade) ne sont pas exécutées si l'état de signal du dernier bit objet de la rotation était 0.

Lorsque vous placez le pavé « Rotation vers gauche de double mot », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

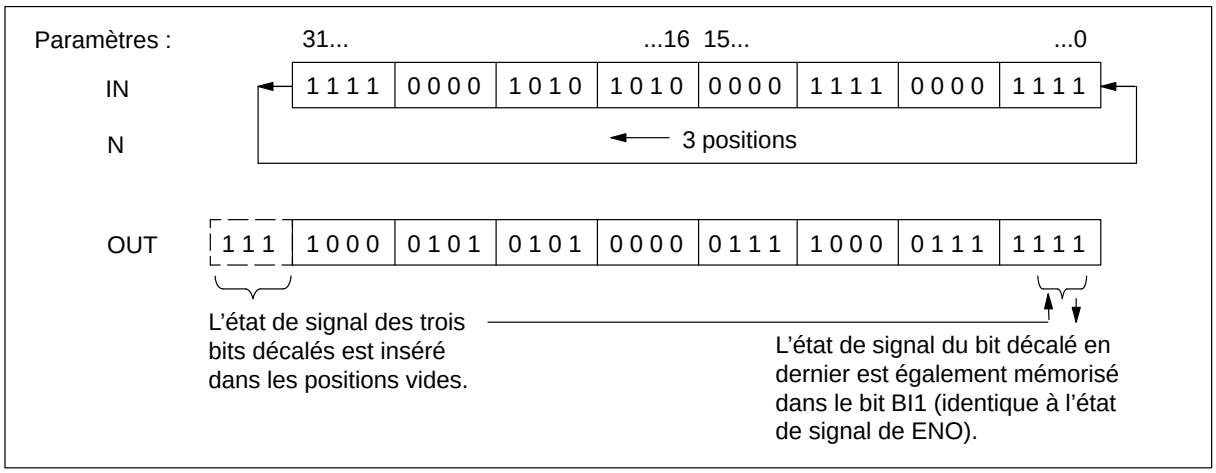
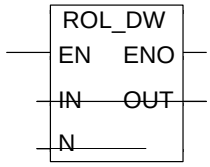
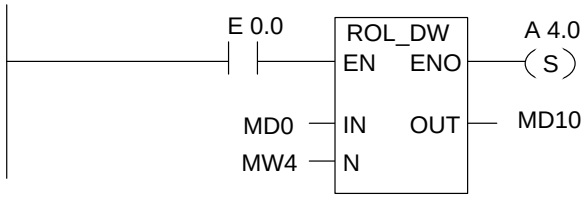

Paramètres :
31...
...16
15...
...0
IN
1 1 1 1
0 0 0 0
1 0 1 0
1 0 1 0
0 0 0 0
1 1 1 1
0 0 0 0
1 1 1 1
N
3 positions
OUT
1 1 1
1 0 0 0
0 1 0 1
0 1 0 1
0 0 0 0
0 1 1 1
1 0 0 0
0 1 1 1
1 1 1 1
L'état de signal des trois bits décalés est inséré dans les positions vides.
L'état de signal du bit décalé en dernier est également mémorisé dans le bit BI1 (identique à l'état de signal de ENO).

Figure 16-10 Rotation des bits de l'entrée IN de trois positions vers la gauche

Tableau 16-7 Rotation vers gauche de double mot : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, L, D	Entrée de validation
	ENO	BOOL	E, A, M, L, D	Sortie de validation
	IN	DWORD	E, A, M, L, D	Valeur objet de la rotation
	N	WORD	E, A, M, L, D	Nombre de bits de rotation
	OUT	DWORD	E, A, M, L, D	Résultat de la rotation



L'opération est exécutée si l'état de signal est 1 à l'entrée E 0.0.

Le double mot de mémento MD0 fait l'objet d'une rotation vers la gauche du nombre de bits précisé dans le mot de mémento MW4.

Le résultat est rangé dans le double mot de mémento MD10. La sortie A 4.0 est mise à 1 si l'état de signal du dernier bit objet de la rotation était 1.

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	x	x	x	–	x	x	x	1

Figure 16-11 Rotation vers gauche de double mot

Rotation vers droite de double mot

L'opération « Rotation vers droite de double mot » est activée si l'état de signal est 1 à l'entrée de validation EN. Elle déclenche la rotation bit par bit vers la droite du contenu entier de l'entrée IN. Le nombre de bits de rotation est indiqué dans l'entrée N dont la valeur doit être comprise entre 0 et 31. Si N est supérieur à 32, le double mot effectue une rotation de ((N–1) modulo 32) + 1) positions vers la droite. Les positions binaires à gauche sont complétées par l'état de signal des bits objet de la rotation. Le résultat de la rotation est rangé dans la sortie OUT.

Lorsque N est différent de zéro, l'opération remet les bits BI0 et DEB du mot d'état à 0. Si elle est exécutée (EN = 1), ENO contient l'état de signal du dernier bit objet de la rotation (identique à BI1 et RLG dans le mot d'état). Ainsi, les fonctions suivant cette opération et qui y sont connectées par ENO (cascade) ne sont pas exécutées si l'état de signal du dernier bit objet de la rotation était 0.

Lorsque vous placez le pavé « Rotation vers droite de double mot », vous devez tenir compte de certaines restrictions (voir paragraphe 6.1).

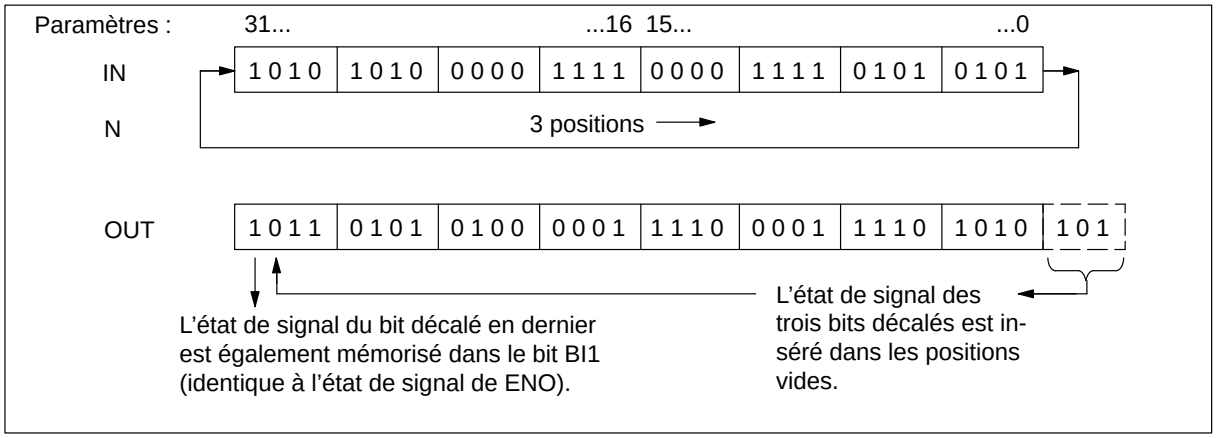
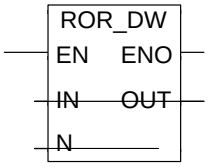
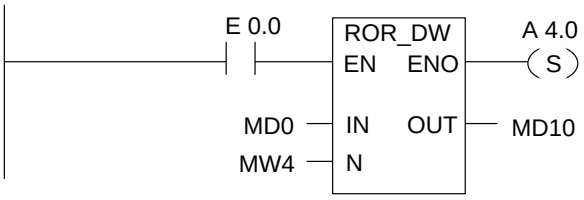

Paramètres :
31...
...16
15...
...0
IN
1 0 1 0
1 0 1 0
0 0 0 0
1 1 1 1
0 0 0 0
1 1 1 1
0 1 0 1
0 1 0 1
N
3 positions
OUT
1 0 1 1
0 1 0 1
0 1 0 0
0 0 0 1
1 1 1 0
0 0 0 1
1 1 1 0
1 0 1 0
1 0 1
L'état de signal du bit décalé en dernier est également mémorisé dans le bit BI1 (identique à l'état de signal de ENO).
L'état de signal des trois bits décalés est inséré dans les positions vides.

Figure 16-12 Rotation des bits de l'entrée IN de trois positions vers la droite

Tableau 16-8 Rotation vers droite de double mot : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
	EN	BOOL	E, A, M, L, D	Entrée de validation
	ENO	BOOL	E, A, M, L, D	Sortie de validation
	IN	DWORD	E, A, M, L, D	Valeur objet de la rotation
	N	WORD	E, A, M, L, D	Nombre de bits de rotation
	OUT	DWORD	E, A, M, L, D	Résultat de la rotation



L'opération est exécutée si l'état de signal est 1 à l'entrée E 0.0.

Le double mot de mémento MD0 fait l'objet d'une rotation vers la droite du nombre de bits précisé dans le mot de mémento MW4.

Le résultat est rangé dans le double mot de mémento MD10. La sortie A 4.0 est mise à 1 si l'état de signal du dernier bit objet de la rotation était 1.

Description des bits du mot d'état

Opération exécutée (EN = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	x	x	x	–	x	x	x	1

Figure 16-13 Rotation vers droite de double mot

Opérations sur blocs de données

17

Contenu de ce chapitre

Paragraphe	Thème	Page
17.1	Ouvrir bloc de données	17-2

17.1 Ouvrir bloc de données

Description

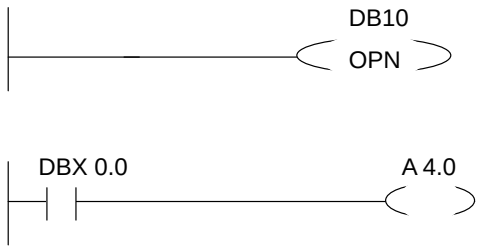
L'opération « Ouvrir bloc de données » ouvre un bloc de données comme bloc de données global (DB) ou bloc de données d'instance (DI). Le numéro du bloc de données est transféré dans le registre DB ou DI. En fonction du contenu du registre, les commandes DB et DI suivantes accèdent aux blocs correspondants.

Tableau 17-1 Ouvrir bloc de données : représentation et paramètres, avec les abréviations SIMATIC

Elément CONT	Paramètre	Type de données	Zone de mémoire	Description
<DB numéro> ou <DI numéro> AUF	Numéro de DB ou de DI	BLOCK_DB	–	La plage de numéros de DB ou de DI dépend de votre CPU.

Tableau 17-2 Ouvrir bloc de données : représentation et paramètres, avec les abréviations internationales

Elément CONT	Paramètre	Type de données	Zone de mémoire	Description
<DB numéro> ou <DI numéro> OPN	Numéro de DB ou de DI	BLOCK_DB	–	La plage de numéros de DB ou de DI dépend de votre CPU.



Le DB10 est le bloc de données ouvert en cours. C'est pourquoi l'interrogation en DBX0.0 se réfère au bit 0 de l'octet de données 0 du bloc de données DB10. L'état de signal de ce bit est affecté à la sortie A 4.0.

Description des bits du mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	–	–	–	–	–	–	–	–	–

L'opération ne lit pas ni ne modifie les bits du mot d'état.

Figure 17-1 Ouvrir bloc de données

Opérations de saut

18

Contenu de ce chapitre

Paragraphe	Thème	Page
18.1	Présentation	18-2
18.2	Saut si 1 (inconditionnel)	18-3
18.3	Saut si 1 (conditionnel)	18-4
18.4	Saut si 0	18-5
18.5	Repère de saut	18-6

18.1 Présentation

Repère de saut comme opérande

L'opérande d'une opération de saut est un repère de saut qui comporte au maximum 4 caractères. Le premier caractère doit être une lettre, les autres caractères pouvant être des lettres ou des chiffres (par exemple, SEG3). Le repère de saut indique la destination où doit sauter le programme.

Le repère de saut doit être indiqué au-dessus de la bobine de saut (voir figure 18-1).

Repère de saut comme destination

Le repère de destination de saut doit se trouver au début du réseau. Pour l'indiquer, sélectionnez REPERE dans la boîte de sélection CONT. Dans le pavé vide qui apparaît, spécifiez ensuite le nom du repère (voir figure 18-1).

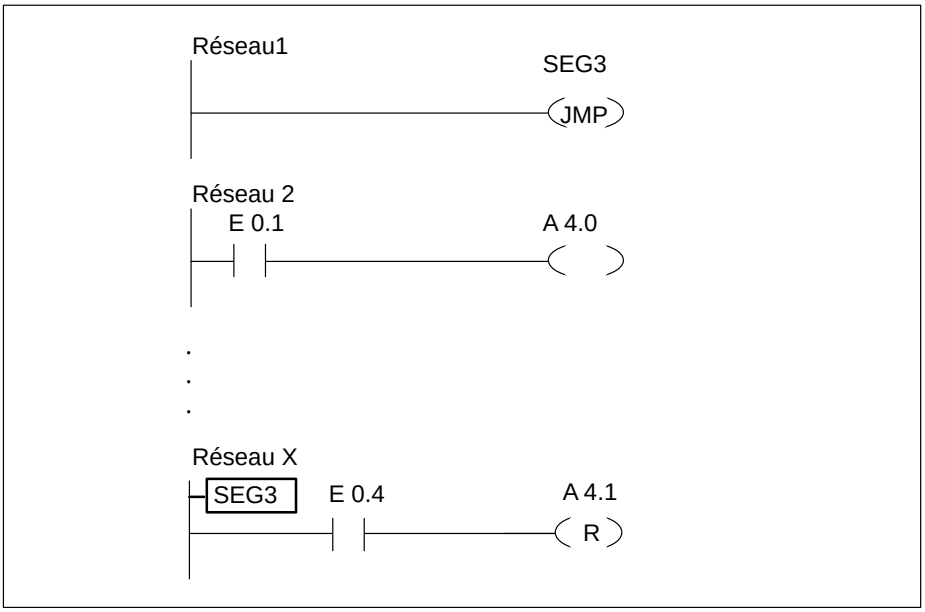


Figure 18-1 Repère de saut comme opérande et comme destination

18.2 Saut si 1 (inconditionnel)

Description

L'opération « Saut si 1 (inconditionnel) » correspond à une opération « Aller à un repère de saut ». Entre la barre d'alimentation gauche et l'opération ne doit figurer aucun autre élément CONT. Aucune des opérations entre l'opération de saut et le repère de saut n'est exécutée.

Vous pouvez utiliser cette opération dans tous les blocs de code, à savoir les blocs d'organisation (OB), les blocs fonctionnels (FB) et les fonctions (FC).

Tableau 18-1 Saut si 1 (inconditionnel) : représentation et paramètres

Elément CONT	Paramètre	Type de données	Zone de mémoire	Description
<opérande> JMP	<repère de saut>	–	–	L'opérande indique le repère vers lequel doit se faire le saut inconditionnel.

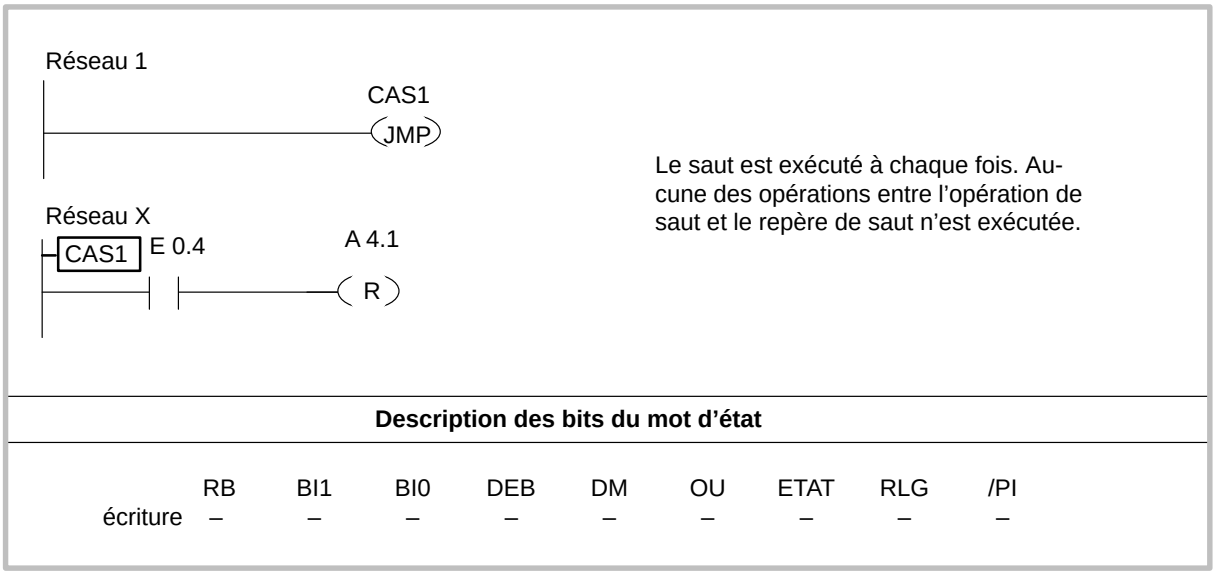

Réseau 1
CAS1
JMP
Réseau X
CAS1
E 0.4
A 4.1
R
Le saut est exécuté à chaque fois. Aucune des opérations entre l'opération de saut et le repère de saut n'est exécutée.
Description des bits du mot d'état
RB BI1 BI0 DEB DM OU ETAT RLG /PI
écriture – – – – – – – – –

Figure 18-2 Saut inconditionnel : aller au repère

18.3 Saut si 1 (conditionnel)

Description

L'opération « Saut si 1 (conditionnel) » correspond à une opération « Aller à un repère de saut » si le RLG est égal à 1. Pour cette opération, utilisez l'élément CONT de l'opération « Saut inconditionnel », mais avec une combinaison en amont. Le saut conditionnel n'est exécuté que si le RLG est égal à 1. Aucune des opérations entre l'opération de saut et le repère de saut n'est exécutée.

Vous pouvez utiliser cette opération dans tous les blocs de code, à savoir les blocs d'organisation (OB), les blocs fonctionnels (FB) et les fonctions (FC).

Tableau 18-2 Saut si 1 (conditionnel) : représentation et paramètres

Elément CONT	Paramètre	Type de données	Zone de mémoire	Description
<opérande> JMP	<repère de saut>	–	–	L'opérande indique le repère vers lequel doit se faire le saut si le RLG est égal à 1.

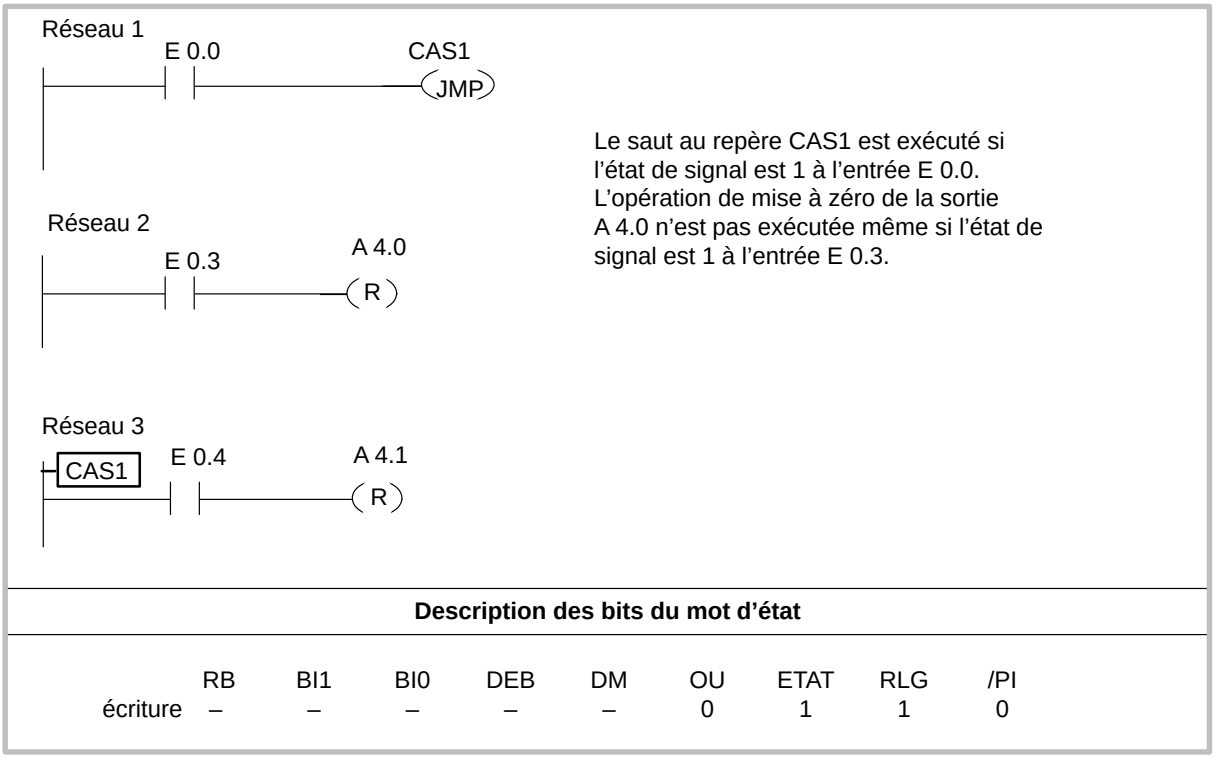

Réseau 1
E 0.0
CAS1
JMP
Le saut au repère CAS1 est exécuté si l'état de signal est 1 à l'entrée E 0.0. L'opération de mise à zéro de la sortie A 4.0 n'est pas exécutée même si l'état de signal est 1 à l'entrée E 0.3.
Réseau 2
E 0.3
A 4.0
R
Réseau 3
CAS1
E 0.4
A 4.1
R
Description des bits du mot d'état
RB BI1 BI0 DEB DM OU ETAT RLG /PI
écriture – – – – – 0 1 1 0

Figure 18-3 Saut conditionnel : saut si 1

18.4 Saut si 0

Description

L'opération « Saut si 0 » correspond à une opération « Aller à un repère de saut » exécutée si le RLG est à 0.

Vous pouvez utiliser cette opération dans tous les blocs de code, à savoir les blocs d'organisation (OB), les blocs fonctionnels (FB) et les fonctions (FC).

Tableau 18-3 Saut si 0 : représentation et paramètres

Elément CONT	Paramètre	Type de données	Zone de mémoire	Description
<opérande> JMP N	<repère de saut>	–	–	L'opérande indique le repère vers lequel doit se faire le saut si le RLG est égal à 0.

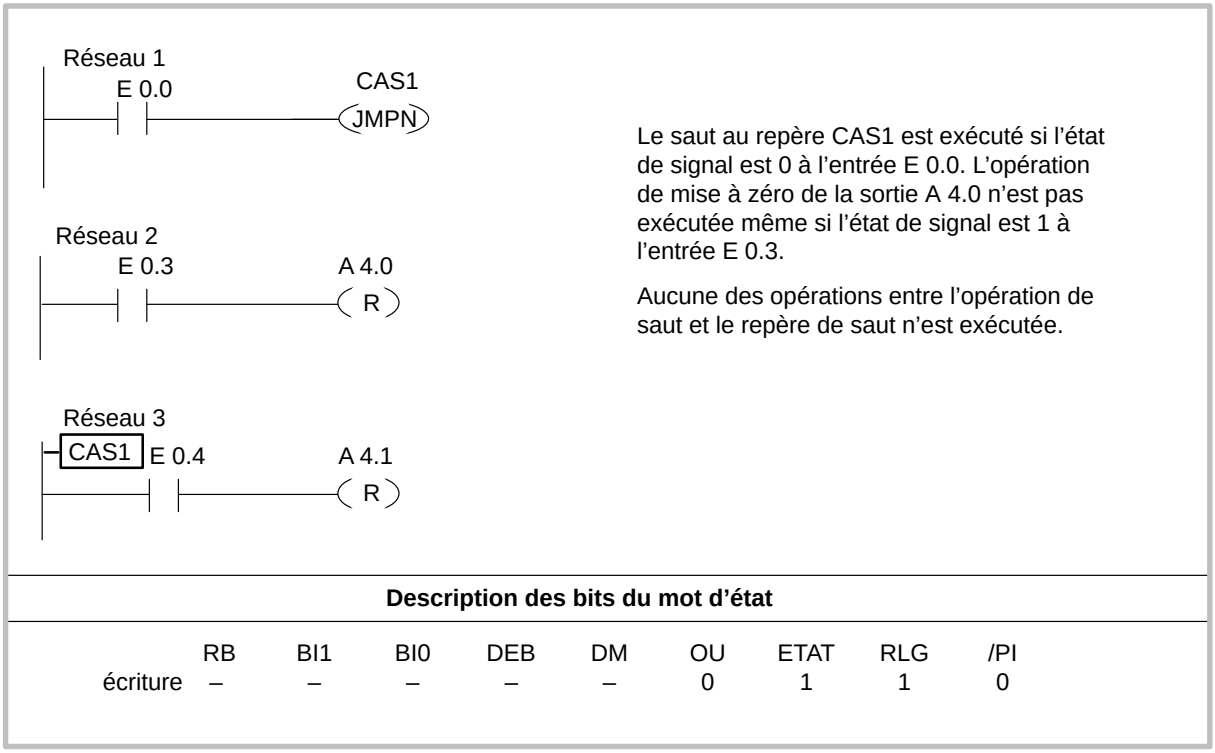

Réseau 1
E 0.0
CAS1
JMPN
Réseau 2
E 0.3
A 4.0
R
Réseau 3
CAS1
E 0.4
A 4.1
R
Le saut au repère CAS1 est exécuté si l'état de signal est 0 à l'entrée E 0.0. L'opération de mise à zéro de la sortie A 4.0 n'est pas exécutée même si l'état de signal est 1 à l'entrée E 0.3.
Aucune des opérations entre l'opération de saut et le repère de saut n'est exécutée.
Description des bits du mot d'état
RB BI1 BI0 DEB DM OU ETAT RLG /PI
écriture – – – – – 0 1 1 0

Figure 18-4 Saut si 0

18.5 Repère de saut

Description

LABEL identifie la destination d'une opération de saut. Il doit exister un repère de saut pour chaque opération —(JMP) ou —(JMPN).

Format	Description
	4 caractères alphanumériques, le premier devant être une lettre.

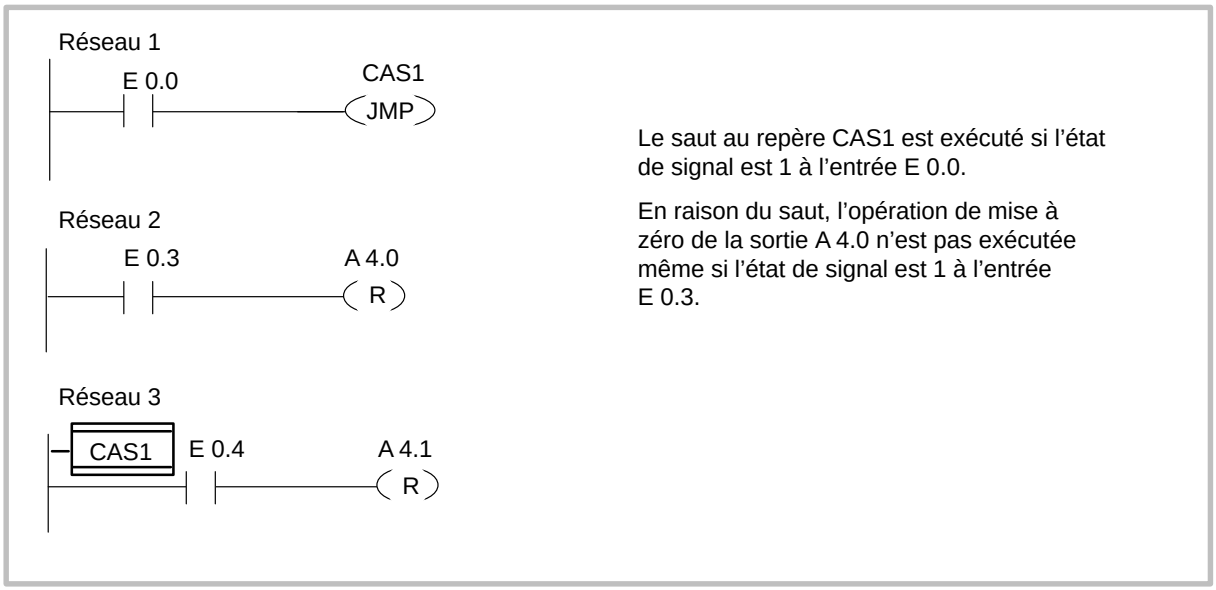

Réseau 1
E 0.0
CAS1
JMP
Réseau 2
E 0.3
A 4.0
R
Réseau 3
CAS1
E 0.4
A 4.1
R
Le saut au repère CAS1 est exécuté si l'état de signal est 1 à l'entrée E 0.0.
En raison du saut, l'opération de mise à zéro de la sortie A 4.0 n'est pas exécutée même si l'état de signal est 1 à l'entrée E 0.3.

Figure 18-5 Repère de saut

Opérations sur bits d’état

19

Contenu de ce chapitre

Paragraphe	Thème	Page
19.1	Présentation	19-2
19.2	Bit d’anomalie « Registre RB »	19-3
19.3	Bits de résultat	19-4
19.4	Bit d’anomalie « Opération illicite »	19-6
19.5	Bit d’anomalie « Débordement »	19-7
19.6	Bit d’anomalie « Débordement mémorisé »	19-9

19.1 Présentation

Description

Les opérations sur bits d'état sont des opérations combinatoires sur bits (voir paragraphe 8.1) qui utilisent les bits du mot d'état (voir paragraphe 6.3). Chacune de ces opérations réagit à l'une des conditions suivantes indiquées par un ou plusieurs bits du mot d'état :

- Le bit de résultat binaire RB est à 1 (son état de signal est égal à 1).
- Le résultat d'une opération arithmétique par rapport à 0 est :
  - supérieur à 0 (>0),
  - inférieur à 0 (<0),
  - supérieur ou égal à 0 (>=0),
  - inférieur ou égal à 0 (<=0),
  - égal à 0 (==0),
  - différent de 0 (<>0).
- Le résultat d'une opération arithmétique est illicite.
- Un débordement s'est produit lors d'une opération arithmétique.

Dans une connexion en série, les opérations sur bits d'état combinent le résultat de leur interrogation d'état de signal avec le résultat logique précédent selon la table de vérité ET (voir paragraphe 6.2 et tableau 6-8). Dans une connexion en parallèle, elles combinent leur résultat avec le RLG précédent selon la table de vérité OU (voir paragraphe 6.2 et tableau 6-9).

Le présent chapitre fournit la représentation internationale et la représentation SIMATIC de l'opération « Bit d'anomalie registre RB » qui interroge l'état de signal du bit RB du mot d'état.

Mot d'état

Le mot d'état est un registre dans la mémoire de votre CPU contenant des bits auxquels vous pouvez accéder dans les opérandes de combinaisons sur bits et sur mots. La figure 19-1 présente la structure du mot d'état. Pour plus d'informations sur les différents bits du mot d'état, reportez-vous au paragraphe 6.3.



Figure 19-1 Structure du mot d'état

Paramètres

Les éléments CONT suivants ne contiennent pas de paramètres pouvant être saisis.

19.2 Bit d'anomalie « Registre RB »

Description

L'opération « Bit d'anomalie Registre RB » permet d'interroger l'état de signal du bit de résultat binaire RB du mot d'état (voir paragraphe 6.3). En série, cette opération combine le résultat de son interrogation avec le résultat logique précédent selon la table de vérité ET (voir paragraphe 6.2 et tableau 6-8). En parallèle, cette opération combine le résultat de son interrogation avec le RLG précédent selon la table de vérité OU (voir paragraphe 6.2 et tableau 6-9).

Représentation et forme inverse de l'élément

La figure 19-2 contient la représentation de « Bit d'anomalie registre RB » et de sa forme inverse. Les éléments sont représentés avec l'abréviation SIMATIC et avec l'abréviation internationale.

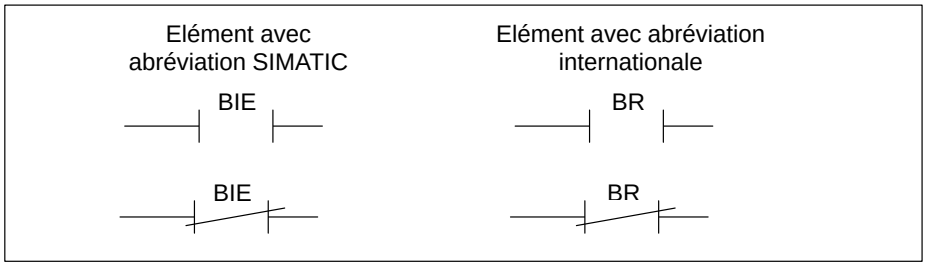
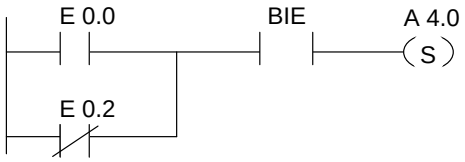


Figure 19-2 Bit d'anomalie « Registre RB » : représentation et forme inverse



La sortie A 4.0 est mise à 1 si l'état de signal est 1 à l'entrée E 0.0 OU 0 à l'entrée E 0.2 et – en plus de ce RLG – si l'état de signal du bit RB est 1.

Description des bits du mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	–	–	–	–	–	x	x	x	1

Figure 19-3 Bit d'anomalie « Registre RB »

19.3 Bits de résultat

Description

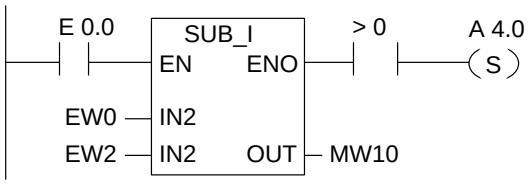
Les opérations « Bits de résultat » permettent de déterminer la relation par rapport à 0 – >0, <0, >=0, <=0, ==0 ou <>0 – du résultat d'une opération arithmétique. (voir tableau 19-1). A cet effet, les bits indicateurs BI1 et BI0 du mot d'état (voir paragraphe 6.3) sont évalués. Si la condition de comparaison précisée dans l'opérande est satisfaite, le résultat de l'interrogation d'état de signal est 1.

En série, cette opération combine le résultat de son interrogation avec le résultat logique précédent selon la table de vérité ET (voir paragraphe 6.2 et tableau 6-8). En parallèle, cette opération combine le résultat de son interrogation avec le RLG précédent selon la table de vérité OU (voir paragraphe 6.2 et tableau 6-9).

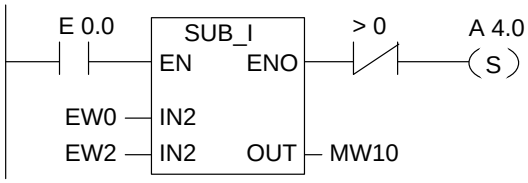
Tableau 19-1 Bits de résultat : représentations et formes inverses

Elément CONT	Description
> 0 > 0	L'opération « Bit de résultat pour supérieur à 0 » détermine si le résultat d'une opération arithmétique est ou non supérieur à 0. Elle interroge la combinaison des bits indicateurs BI1 et BI0 du mot d'état afin de déterminer la relation du résultat par rapport à 0.
< 0 < 0	L'opération « Bit de résultat pour inférieur à 0 » détermine si le résultat d'une opération arithmétique est ou non inférieur à 0. Elle interroge la combinaison des bits indicateurs BI1 et BI0 du mot d'état afin de déterminer la relation du résultat par rapport à 0.
> = 0 > = 0	L'opération « Bit de résultat pour supérieur ou égal à 0 » détermine si le résultat d'une opération arithmétique est ou non supérieur ou égal à 0. Elle interroge la combinaison des bits indicateurs BI1 et BI0 du mot d'état afin de déterminer la relation du résultat par rapport à 0.
< = 0 < = 0	L'opération « Bit de résultat pour inférieur ou égal à 0 » détermine si le résultat d'une opération arithmétique est ou non inférieur ou égal à 0. Elle interroge la combinaison des bits indicateurs BI1 et BI0 du mot d'état afin de déterminer la relation du résultat par rapport à 0.
== 0 == 0	L'opération « Bit de résultat pour égal à 0 » détermine si le résultat d'une opération arithmétique est ou non égal à 0. Elle interroge la combinaison des bits indicateurs BI1 et BI0 du mot d'état afin de déterminer la relation du résultat par rapport à 0.
< > 0 < > 0	L'opération « Bit de résultat pour différent de 0 » détermine si le résultat d'une opération arithmétique est ou non différent de 0. Elle interroge la combinaison des bits indicateurs BI1 et BI0 du mot d'état afin de déterminer la relation du résultat par rapport à 0.

1)



2)



L'opération SUB_I est activée si l'état de signal est 1 à l'entrée E 0.0. Si la valeur du mot d'entrée EW0 est supérieure à celle du mot d'entrée EW2, le résultat de l'opération arithmétique EW0 - EW2 est supérieur à 0.
Si EN est à 1 et qu'une erreur se produit lors de l'exécution de l'opération SUB_I, ENO est mis à 0.

1) La sortie A 4.0 est mise à 1 si l'opération arithmétique s'exécute sans erreur et si le résultat est supérieur à 0. Si l'état de signal est 0 à l'entrée E 0.0 (désactivée), EN et ENO sont tous deux mis à 0.

2) La sortie A 4.0 est mise à 1 si l'opération arithmétique s'exécute sans erreur et si le résultat est inférieur ou égal à 0. Si l'état de signal est 0 à l'entrée E 0.0 (désactivée), EN et ENO sont tous deux mis à 0.

Description des bits du mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	–	–	–	–	–	x	x	x	1

Figure 19-4 Bit de résultat pour supérieur à 0 et Bit de résultat pour supérieur à 0 inverse

19.4 Bit d'anomalie « Opération illicite »

Description

L'opération « Bit d'anomalie Opération illicite » permet de déterminer si le résultat d'une opération arithmétique à virgule flottante est ou non illicite, c'est-à-dire si l'une des valeurs de l'opération arithmétique n'est pas un nombre à virgule flottante autorisé. A cet effet, les bits indicateurs BI1 et BI0 du mot d'état (voir paragraphe 6.3) sont évalués. Si le résultat de l'opération arithmétique est illicite (UO), l'interrogation de l'état de signal fournit un résultat égal à 1. Si la combinaison dans BI1 et BI0 ne signifie pas illicite, le résultat de l'interrogation de l'état de signal est 0.

En série, cette opération combine le résultat de son interrogation avec le résultat logique (RLG, voir paragraphe 6.3) précédent selon la table de vérité ET (voir paragraphe 6.2 et tableau 6-8). En parallèle, cette opération combine le résultat de son interrogation avec le RLG précédent selon la table de vérité OU (voir paragraphe 6.2 et tableau 6-9).

Représentation et forme inverse de l'élément

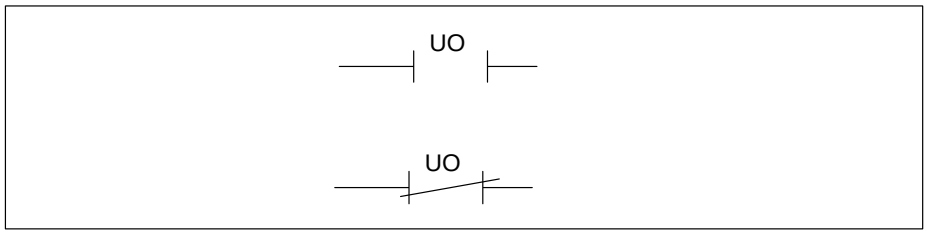
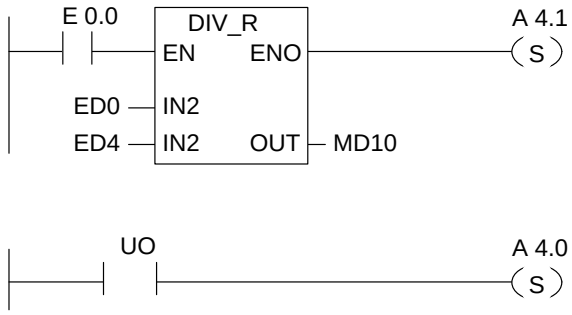


Figure 19-5 Bit d'anomalie « Opération illicite » : représentation et forme inverse



L'opération DIV_R est activée si l'état de signal est 1 à l'entrée E 0.0. Si la valeur du double mot d'entrée ED0 ou ED4 n'est pas un nombre à virgule flottante correct, l'opération arithmétique à virgule flottante est illicite.

Si EN est à 1 (activée) et qu'une erreur se produise lors de l'exécution de l'opération DIV_R, ENO est mis à 0.

La sortie A 4.0 est mise à 1 si l'opération DIV_R est exécutée mais que l'une des valeurs dans l'opération arithmétique n'est pas un nombre à virgule flottante correct. Si l'état de signal est 0 à l'entrée E 0.0 (désactivée), EN et ENO sont tous deux mis à 0.

Description des bits du mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	–	–	–	–	–	x	x	x	1

Figure 19-6 Bit d'anomalie « Opération illicite »

19.5 Bit d'anomalie « Débordement »

Description

L'opération « Bit d'anomalie Débordement » permet de détecter un débordement dans la dernière opération arithmétique traitée. Si, après une opération arithmétique exécutée par le système, le résultat se situe hors de la plage négative ou positive autorisée, le bit de débordement (DEB) du mot d'état (voir paragraphe 6.3) est mis à 1. L'opération « Bit d'anomalie Débordement » interroge l'état de signal de ce bit. Lorsque l'erreur est corrigée, ce bit est mis à 0.

En série, cette opération combine le résultat de son interrogation avec le résultat logique précédent selon la table de vérité ET (voir paragraphe 6.2 et tableau 6-8). En parallèle, cette opération combine le résultat de son interrogation avec le RLG précédent selon la table de vérité OU (voir paragraphe 6.2 et tableau 6-9).

Représentation et forme inverse de l'élément

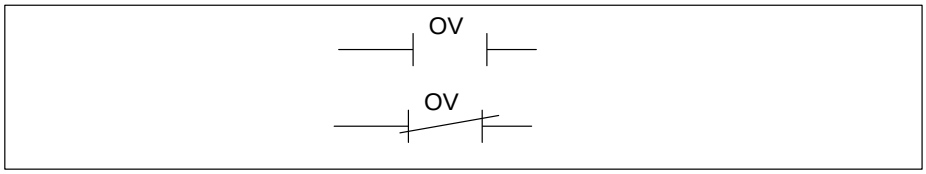
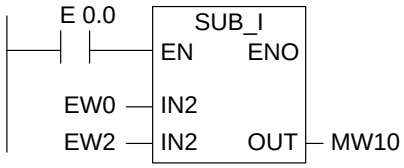


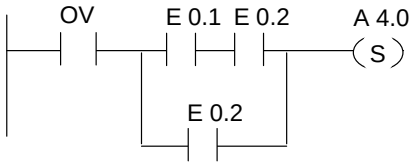
Figure 19-7 Bit d'anomalie « Débordement » : représentation et forme inverse

Réseau 1



L'opération SUB_I est activée si l'état de signal est 1 à l'entrée E 0.0. Si le résultat de l'opération arithmétique EW0 - EW2 est hors de la plage autorisée pour un nombre entier, le bit DEB du mot d'état est mis à 1.

Réseau 2



Le résultat d'une interrogation d'état de signal pour DEB est égal à 1. La sortie A 4.0 est mise à 1 si l'interrogation de DEB donne 1 et que le RLG du second réseau soit égal à 1 (c'est-à-dire si le RLG juste avant la sortie A 4.0 est 1).

Si l'état de signal est 0 à l'entrée E 0.0 (désactivée), EN et ENO sont tous deux mis à 0. Si EN est à 1 (activée) et que le résultat de l'opération arithmétique se situe hors de la plage autorisée, ENO est mis à 0.

Remarque : L'interrogation de DEB n'est nécessaire qu'en raison de la présence de plusieurs réseaux. Sinon, il suffit de considérer la sortie de validation ENO de l'opération arithmétique qui est à 0 si le résultat ne se situe pas dans la plage autorisée.

Description des bits du mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	–	–	–	–	–	x	x	x	1

Figure 19-8 Bit d'anomalie « Débordement »

19.6 Bit d'anomalie « Débordement mémorisé »

Description

L'opération « Bit d'anomalie Débordement mémorisé » permet de détecter un débordement mémorisé dans une opération arithmétique. Si, après une opération arithmétique exécutée par le système, le résultat se situe hors de la plage négative ou positive autorisée, le bit DM du mot d'état (voir paragraphe 6.3) est mis à 1. L'opération « Bit d'anomalie Débordement mémorisé » interroge l'état de signal de ce bit. Contrairement au bit DEB (débordement), le bit DM (débordement mémorisé) reste à 1 après correction de l'erreur (voir paragraphe 19.5).

En série, cette opération combine le résultat de son interrogation avec le résultat logique précédent selon la table de vérité ET (voir paragraphe 6.2 et tableau 6-8). En parallèle, cette opération combine le résultat de son interrogation avec le RLG précédent selon la table de vérité OU (voir paragraphe 6.2 et tableau 6-9).

Représentation et forme inverse de l'élément

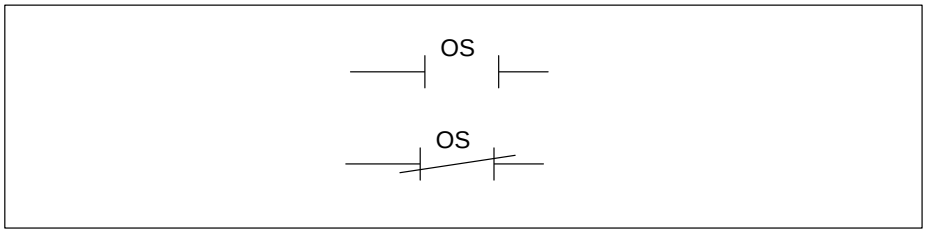
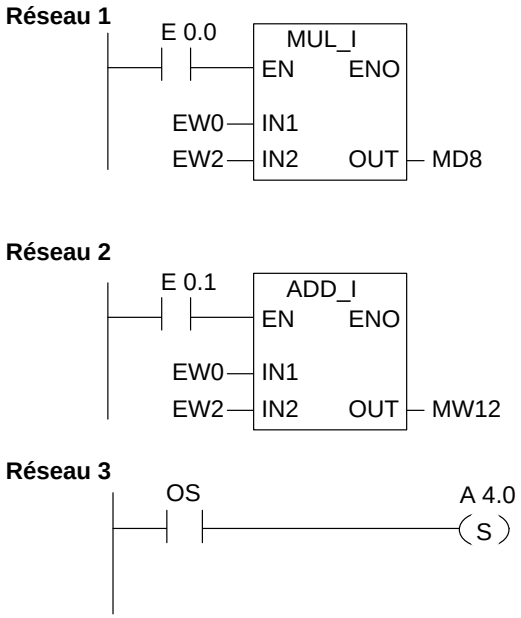


Figure 19-9 Bit d'anomalie « Débordement mémorisé » : représentation et forme inverse



L'opération MUL_I est activée si l'état de signal est 1 à l'entrée E 0.0. L'opération ADD_I est activée si l'état de signal est 1 à l'entrée E 0.1. Si le résultat de l'une de ces opérations arithmétiques est hors de la plage autorisée pour un nombre entier, le bit DM du mot d'état est mis à 1.

Le résultat d'une interrogation d'état de signal pour DM est égal à 1 et la sortie A 4.0 est mise à 1.

Dans le réseau 1, si l'état de signal est 0 à l'entrée E 0.0 (désactivée), EN et ENO sont tous deux mis à 0. Si EN est à 1 (activée) et que le résultat de l'opération arithmétique se situe hors de la plage autorisée, ENO est mis à 0.

Dans le réseau 2, si l'état de signal est 0 à l'entrée E 0.1 (désactivée), EN et ENO sont tous deux mis à 0. Si EN est à 1 (activée) et que le résultat de l'opération arithmétique se situe hors de la plage autorisée, ENO est mis à 0.

Remarque :
L'interrogation de DM n'est nécessaire qu'en raison de la présence de plusieurs réseaux. Sinon, il suffit de connecter la sortie de validation ENO de la première opération arithmétique à l'entrée de validation EN de la seconde (connexion en cascade).

Description des bits du mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	–	–	–	–	–	x	x	x	1

Figure 19-10 Bit d'anomalie « Débordement mémorisé »

Opérations de gestion d’exécution de programme

20

Contenu de ce chapitre

Paragraphe	Thème	Page
20.1	Appeler FC/SFC sans paramètre	20-2
20.2	Appeler FB, FC, SFB, SFC et multi-instances	20-4
20.3	Retour	20-7
20.4	Opérations du relais de masquage	20-8
20.5	Activer/désactiver relais de masquage	20-9
20.6	Relais de masquage en fonction/hors fonction	20-12

20.1 Appeler FC/SFC sans paramètre

Description

L'opération « Appeler FC/SFC sans paramètre » permet d'appeler une fonction (FC) ou une fonction système (SFC) qui n'a pas de paramètre. L'appel est conditionnel ou inconditionnel selon le lien précédent (voir exemple dans la figure 20-1).

Dans le cas d'un appel conditionnel, vous ne pouvez pas indiquer le type de données BLOCK_FC pour les paramètres d'une fonction (FC). Par contre, dans un bloc fonctionnel (FB), vous pouvez indiquer BLOCK_FC comme type de paramètre.

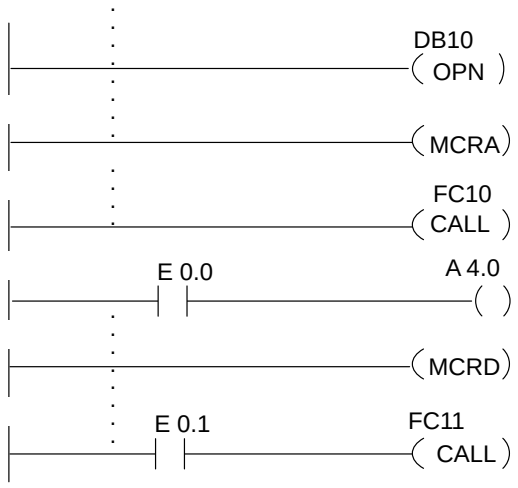
Un appel conditionnel n'est exécuté que si le résultat logique RLG est à 1. Si un appel conditionnel n'est pas exécuté, le RLG est à 0 après l'opération d'appel. Si l'opération a lieu, elle fonctionne comme suit :

- Elle sauvegarde l'adresse de retour au bloc appelant.
- Elle sauvegarde les sélecteurs des deux blocs de données en cours (DB et DB d'instance).
- Elle change la zone de données locales en cours en zone de données locales précédente.
- Elle empile le bit MA (bit MCR actif) dans la pile des blocs.
- Elle crée la nouvelle zone de données locales pour la fonction ou la fonction système appelée.

Ensuite, le programme poursuit le traitement dans le bloc appelé. Pour plus d'informations sur la transmission des paramètres, reportez-vous au manuel de programmation **/120/**.

Tableau 20-1 Appeler FC/SFC sans paramètre : représentation et paramètre

Elément CONT	Paramètre	Type de données	Zone de mémoire	Description
numéro CALL	Numéro	BLOCK_FC	–	Numéro de la fonction ou fonction système (par exemple, FC10 ou SFC59). Les SFC disponibles dépendent de votre CPU. Dans le cas d'un appel conditionnel, vous ne pouvez indiquer le type de données BLOCK_FC que pour les paramètres d'un bloc fonctionnel (FB), et non d'une fonction (FC).



Si l'appel inconditionnel de FC10 est exécuté, l'opération CALL fonctionne comme suit :

- Elle sauvegarde l'adresse de retour au FB en cours.
- Elle sauvegarde les sélecteurs pour DB10 et pour le bloc de données d'instance du FB.
- Elle empile le bit MA, mis à 1 par l'opération —(MCRA), dans la pile des blocs et remet ce bit à 0 pour la fonction FC10 appelée.

Le traitement du programme se poursuit dans FC10. Si vous voulez utiliser la fonction MCR dans FC10, vous devez l'y réactiver. A la fin de FC10, le traitement du programme revient au FB appelant. Le bit MA est restauré. Le DB10 et le bloc de données d'instance du FB utilisateur redeviennent les DB en cours, quels qu'aient été les DB utilisés par FC10.

Le programme se poursuit avec l'opération suivante, dans cet exemple l'affectation de l'état de signal en E 0.0 à la sortie A 4.0. L'appel de FC11 étant conditionnel, il n'est exécuté que si l'état de signal en E 0.1 est 1. S'il est exécuté, il se déroule comme l'appel de FC10.

Description des bits du mot d'état

Appel inconditionnel

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	–	–	–	–	0	0	1	–	0

Appel conditionnel

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	–	–	–	–	0	0	1	1	0

Figure 20-1 Appeler FC/SFC sans paramètre

20.2 Appeler FB, FC, SFB, SFC et multi-instances

Description

Vous pouvez appeler des blocs fonctionnels (FB), des fonctions (FC), des blocs fonctionnels système (SFB), des fonctions système (SFC) et des multi-instances en les sélectionnant dans la boîte de dialogue « Eléments de programme ». Ils se trouvent à la fin de la liste des familles d'opérations, sous les noms suivants :

- Blocs FB
- Blocs FC
- Blocs SFB
- Blocs SFC
- Multi-instances
- Bibliothèques

Lorsque vous sélectionnez l'un de ces blocs, le pavé correspondant apparaît à l'écran, avec le numéro ou le mnémonique de la fonction ou du bloc fonctionnel, ainsi que les paramètres correspondants.

Le bloc que vous appelez doit avoir été compilé préalablement et doit déjà exister dans votre fichier programme, dans la bibliothèque ou dans la CPU.

Si l'opération « Appeler FB, FC, SFB, SFC et multi-instances sous forme de pavé » a lieu, elle fonctionne comme suit :

- Elle sauvegarde l'adresse de retour au bloc appelant.
- Elle sauvegarde les sélecteurs des deux blocs de données en cours (DB et DI).
- Elle change la zone de données locales en cours en zone de données locales précédente.
- Elle empile le bit MA (bit MCR actif) dans la pile des blocs.
- Elle crée la nouvelle zone de données locales pour la fonction FC ou SFC appelée.

Nota

Lorsque les registres DB et DI sont sauvegardés, il se peut qu'ils ne pointent pas sur les blocs de données que vous avez ouverts. En raison des mécanismes de copie lors de la transmission de paramètres, tout particulièrement avec les blocs fonctionnels, le compilateur écrase parfois le registre DB. De plus amples informations à ce sujet sont données dans le manuel de programmation **/234/**.

Ensuite, le programme poursuit le traitement dans le bloc appelé.

Sortie de validation

La sortie de validation (ENO) d'un pavé CONT correspond au bit RB du mot d'état (voir paragraphe 6.3). Lorsque vous écrivez – indifféremment en LIST ou en CONT – un bloc fonctionnel ou une fonction que vous voulez appeler en CONT, vous devez tenir compte du bit RB. Utilisez l'opération SAVE (en LIST) ou la bobine —(SAVE) (en CONT) pour mémoriser le RLG dans le bit RB, en tenant compte des critères suivants :

- Au cas où le FB ou la FC sont exécutés sans erreur, mémorisez un RLG égal à 1 dans le bit RB
- Au cas où le FB ou la FC sont exécutés avec erreur, mémorisez un RLG égal à 0 dans le bit RB

Programmez ces opérations à la fin du FB ou de la FC afin qu'elles soient les dernières opérations traitées dans le bloc.



Attention

Le bit RB risque d'être remis involontairement à 0.

Lorsque vous écrivez des FB et FC en schéma à contacts sans tenir compte des recommandations ci-avant, un FB ou une FC risque d'écraser le bit RB d'un autre FB ou d'une autre FC.

Pour éviter cette erreur, mémorisez le RLG à la fin de chaque FB ou FC, comme décrit précédemment.

Effets de l'appel sur les bits du mot d'état

La figure 20-2 montre les effets d'un appel conditionnel et d'un appel inconditionnel d'un bloc sur les bits du mot d'état (voir paragraphe 6.3).

		RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
Conditionnel :	écriture	x	–	–	–	0	0	1	1	x
Inconditionnel :	écriture	–	–	–	–	0	0	1	–	x

Figure 20-2 Effets d'un appel de bloc sur les bits du mot d'état

Paramètres

Les paramètres du bloc appelé s'affichent dans la représentation CONT. Selon le type de bloc, vous devez fournir les paramètres suivants :

- Dans le cas d'une fonction (FC), vous devez fournir des paramètres effectifs pour tous les paramètres formels.
- Dans le cas d'un bloc fonctionnel (FB), il n'est pas indispensable de fournir des paramètres effectifs pour tous les paramètres formels. Vous devez cependant fournir un bloc de données d'instance (DB d'instance) au FB. Si l'un des paramètres formels n'a pas été pourvu d'un paramètre effectif, le FB utilise les valeurs contenues dans son bloc de données d'instance.
- Dans le cas des multi-instances, vous ne devez plus indiquer le DB d'instance : le pavé appelé contient déjà le numéro de DB correspondant (pour la déclaration des multi-instances, voir le paragraphe 3.5).

Dans le cas des paramètres INOUT complexes et des paramètres de type POINTER et ARRAY, vous devez fournir un paramètre effectif, du moins lors du premier appel du bloc.

Lorsque vous fournissez un paramètre effectif lors de l'appel d'un bloc fonctionnel, veillez à ce qu'il soit de type identique au paramètre formel correspondant.

Pour obtenir des informations sur la manière de programmer une fonction ou d'utiliser ses paramètres, reportez-vous au manuel de programmation **/234/**.

Le tableau 20-2 contient la représentation de « Appeler FB, FC, SFB, SFC et multi-instances » et présente les paramètres communs à la représentation de tous ces appels de blocs. Lorsque vous appelez un bloc à partir de la boîte de dialogue, son numéro apparaît automatiquement comme titre du bloc (numéro du FB, de la FC, du SFB ou de la SFC, par exemple FC10).

Tableau 20-2 Appeler FB, FC, SFB, SFC et multi-instances : représentation et paramètres

Pavé CONT	Paramètre	Type de données	Zone de mémoire	Description
DB n° N° de bloc EN ENO IN OUT IN/OUT	n°	BLOCK_DB	–	Numéro du bloc de données d'instance que vous devez fournir lors de l'appel de FB.
	EN	BOOL	E, A, M, D, L	Entrée de validation
	ENO	BOOL	E, A, M, D, L	Sortie de validation

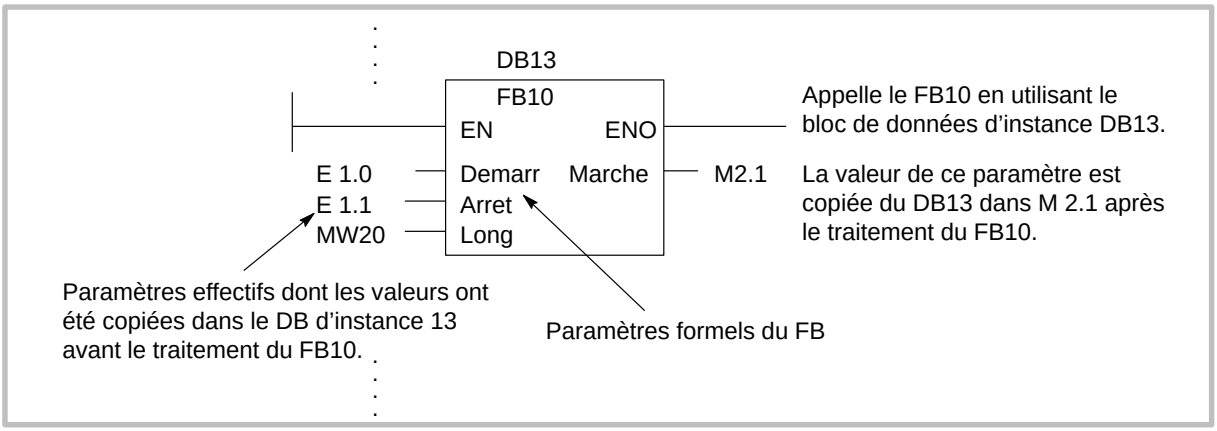

DB13
FB10
EN
ENO
E 1.0
E 1.1
MW20
Demarr
Arret
Long
Marche
M2.1
Appelle le FB10 en utilisant le bloc de données d'instance DB13.
La valeur de ce paramètre est copiée du DB13 dans M 2.1 après le traitement du FB10.
Paramètres effectifs dont les valeurs ont été copiées dans le DB d'instance 13 avant le traitement du FB10.
Paramètres formels du FB

Figure 20-3 Appeler FB sous forme de pavé

20.3 Retour

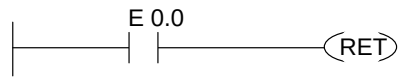
Description

L'opération « Retour » permet de quitter des blocs conditionnellement. Elle sauvegarde le RLG dans le bit de résultat binaire RB du mot d'état.

En cas d'abandon conditionnel d'un bloc, l'état de signal du RLG et le bit RB du bloc dans lequel le programme poursuit le traitement sont mis à 1.

Tableau 20-3 Retour : représentation

Elément CONT	Paramètre	Type de données	Zone de mémoire	Description
—(RET)	Néant	–	–	–



Le bloc est abandonné si l'état de signal est 1 à l'entrée E 0.0. Le bit RB du mot d'état prend alors le même état de signal que l'entrée E 0.0 (= 1).

Description des bits du mot d'état

Retour conditionnel (si RLG = 1) :

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	x	–	–	–	x	0	1	1	0

Figure 20-4 Retour

20.4 Opérations du relais de masquage

Définition du relais de masquage

Le relais de masquage (MCR) (en anglais « Master Control Relay », voir également paragraphe 20.5) est utilisé dans les schémas à contacts de relais pour activer et désactiver le flux d'énergie (trajet du courant). Un trajet de courant désactivé correspond à une séquence d'opérations qui inscrit la valeur zéro au lieu de la valeur calculée, ou à une séquence d'opérations qui laisse la valeur existante inchangée en mémoire. Les fonctions déclenchées par les opérations représentées dans le tableau 20-4 dépendent du relais de masquage MCR.

Les opérations « Sortie » et « Connecteur » inscrivent la valeur 0 en mémoire si le MCR est égal à 0. Les opérations « Mettre à 1 » et « Mettre à 0 » laissent la valeur existante inchangée (voir tableau 20-5).

Tableau 20-4 Opérations influencées par une zone MCR

Représentation ou nom du pavé	Nom de l'opération	Paragraphe dans ce manuel
—(#)—	Connecteur	8.5
—( )—	Sortie	8.4
—(S)—	Mettre à 1	8.8
—(R)—	Mettre à 0	8.9
SR	Bascule mise à 1, mise à 0	8.22
RS	Bascule mise à 0, mise à 1	8.23
MOVE	Affecter valeur	14.1

Tableau 20-5 Opérations dépendant du MCR et réactions de ces opérations à l'état de signal du MCR

Etat de signal du MCR	Sortie ou connecteur —( )— —(#)—	Mettre à 1 ou mettre à 0 —(S)— —(R)— SR RS	Affecter valeur MOVE
0	Ecrit « 0 ». (imite un relais qui passe à l'état de repos en cas de coupure de courant)	N'écrit pas. (imite un relais de maintien qui reste à l'état actuel en cas de coupure de courant)	Ecrit « 0 ». (imite un composant qui fournit la valeur 0 en cas de coupure de courant)
1	Exécution normale	Exécution normale	Exécution normale

20.5 Activer/désactiver relais de masquage

Activer relais de masquage

L'opération « Activer relais de masquage » active la fonction de dépendance entre le relais de masquage (MCR : Master Control Relay) et les commandes suivantes. Vous pouvez, après cette commande, faire appel aux opérations « Relais de masquage en fonction » et « Relais de masquage hors fonction » pour programmer des zones MCR (voir paragraphe 20.6). Lorsque votre programme active une zone MCR, toutes les actions MCR dépendent du contenu de la pile MCR (voir figure 20-5).

Tableau 20-6 Activer relais de masquage : représentation

Elément CONT	Paramètre	Type de données	Zone de mémoire	Description
—(MCRA)	Néant	–	–	Active la fonction MCR.

Désactiver relais de masquage

L'opération « Désactiver relais de masquage » désactive la fonction de dépendance entre le relais de masquage et les commandes suivantes. Après cette opération, vous ne pouvez plus programmer de zones MCR. Lorsque votre programme désactive une zone MCR, le courant circule toujours dans le relais de masquage, indépendamment des entrées figurant dans la pile MCR.

Tableau 20-7 Désactiver relais de masquage : représentation

Elément CONT	Paramètre	Type de données	Zone de mémoire	Description
—(MCRD)	Néant	–	–	Désactive la fonction MCR.

La pile MCR et le bit MA qui commande la fonction de dépendance de la pile se réfèrent à chaque fois à un niveau de séquence. C'est la raison pour laquelle vous devez les sauvegarder puis les appeler à chaque changement de niveau. Au début de chaque niveau de séquence, des valeurs par défaut sont définies pour la pile MCR et le bit MA (bits de saisie MCR de 1 à 8 mis à 1, pointeur de pile MCR = 0 et bit MA = 0).

La pile MCR est transmise de bloc en bloc. A chaque appel de bloc, le bit MA est sauvegardé et remis à 0. En fin de bloc, le bit MA est restauré.

Le relais de masquage peut être réalisé de façon à optimiser les temps d'exécution nécessaires aux CPU génératrices de code. L'optimisation s'explique par le fait que la dépendance par rapport au relais de masquage n'est pas transmise au bloc, mais qu'elle doit être activée explicitement par une opération MCRA. La CPU génératrice de code reconnaît l'opération MCRA et génère le code supplémentaire nécessaire à l'analyse de la pile MCR jusqu'à ce qu'elle reconnaisse une opération MCRD ou la fin du bloc. Le temps d'exécution n'est donc pas allongé pour les opérations programmées hors de la zone délimitée par MCRA et MCRD.

Dans votre programme, vous ne devez jamais utiliser l'opération MCRA sans l'opération MCRD.

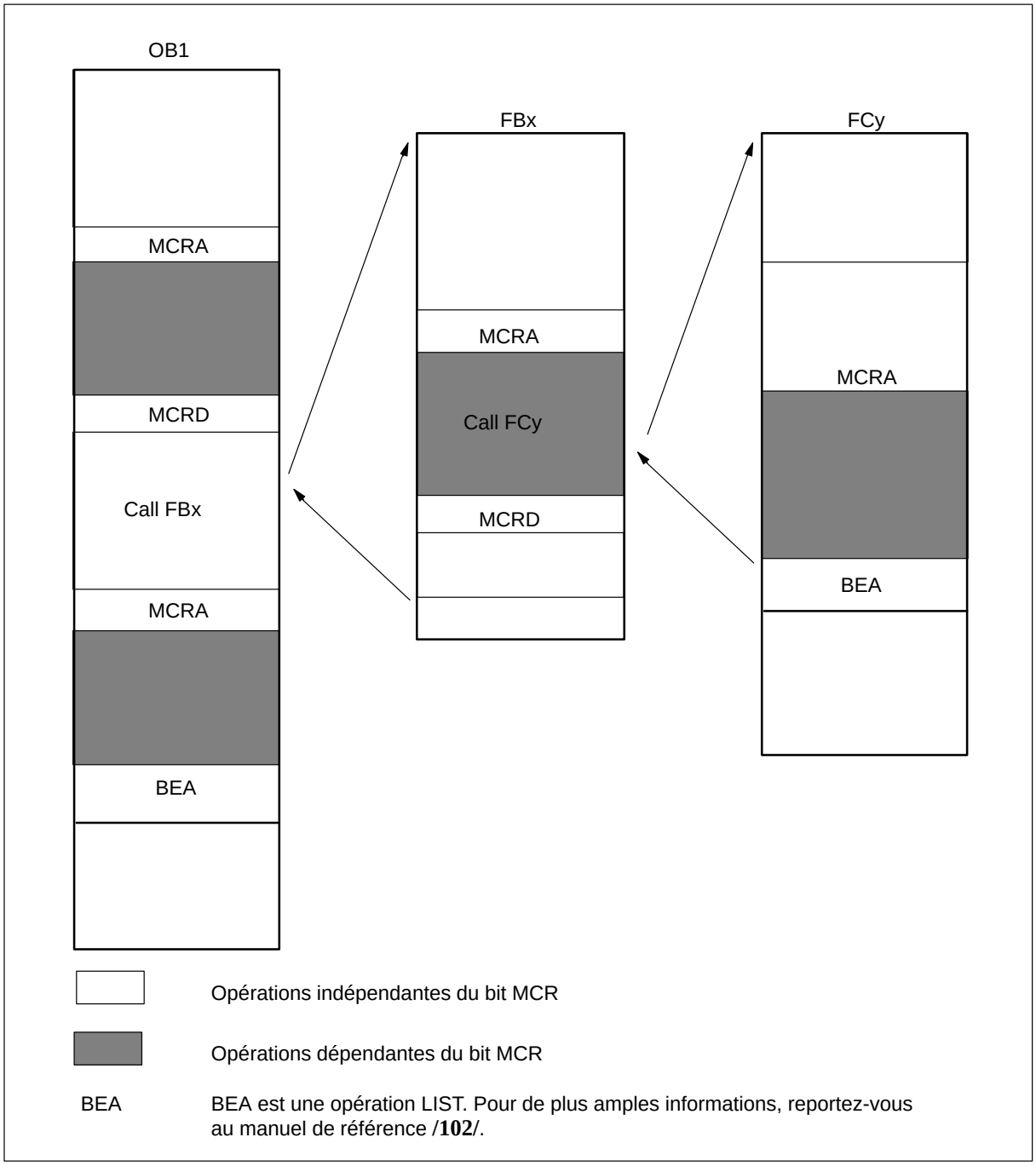
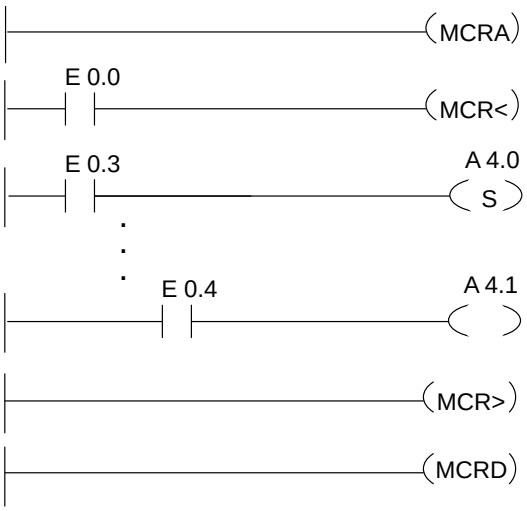


Figure 20-5 Activer et désactiver un relais de masquage

Contrairement aux opérations programmées entre MCRA et MCRD, les opérations programmées hors de la séquence MCRA-MCRD ne dépendent pas de l'état du signal du bit MCR. S'il manque une opération MCRD, les opérations programmées entre MCRA et BEA (opération LIST) dépendent du bit MCR. Pour de plus amples informations, reportez-vous au manuel **/232/**.



L'opération —(MCRA) active la fonction MCR jusqu'au prochain MCRD. Les opérations entre —(MCR<) et —(MCR>) sont traitées, en fonction du bit MA, en l'occurrence E 0.0 :

- Si l'état de signal de l'entrée E 0.0 est à 1 :
  - la sortie A 4.0 est mise à 1 si l'état de signal est 1 à l'entrée E 0.3 ;
  - la sortie A 4.0 reste inchangée si l'état de signal est 0 à l'entrée E 0.3 ;
  - l'état de signal à l'entrée E 0.4 est affecté à la sortie A 4.1.
- Si l'état de signal de l'entrée E 0.0 est à 0 :
  - la sortie A 4.0 reste inchangée quel que soit l'état de signal à l'entrée E 0.3 ;
  - la sortie A 4.1 est à 0 quel que soit l'état de signal à l'entrée E 0.4.

Description des bits du mot d'état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	–	–	–	–	–	–	–	–	–

Figure 20-6 Activer/désactiver relais de masquage

La dépendance des fonctions (FC) et des blocs fonctionnels (FB) par rapport au relais de masquage (MCR) doit être programmée dans les blocs par vous-même. Appeler une fonction ou un bloc fonctionnel à partir d'une séquence MCRA-MCRD n'implique pas automatiquement une relation de dépendance entre les instructions programmées dans cette séquence et le bit MCR. Pour ce faire, vous devez utiliser l'opération MCRA dans le bloc appelé.



Attention

Risques de blessures et de dégâts matériels

N'utilisez jamais l'opération MCR comme fonction d'ARRET D'URGENCE ni comme dispositif de sécurité pour les personnes.

Le MCR ne peut en aucun cas remplacer un relais de masquage câblé.

20.6 Relais de masquage en fonction/hors fonction

Relais de masquage en fonction

L'opération « Relais de masquage en fonction » (MCR<) empile le résultat logique RLG dans la pile MCR, ce qui ouvre une zone MCR. Les opérations présentées dans le tableau 20-4 sont influencées par le RLG qui est empilé dans la pile MCR à l'ouverture d'une zone MCR. La pile MCR, qui fonctionne selon le principe « dernier entré, premier sorti », peut contenir jusqu'à huit entrées. Si elle est pleine, l'opération « Relais de masquage en fonction » provoque une erreur de pile MCR (MCRF).

Tableau 20-8 Relais de masquage en fonction : représentation

Elément CONT	Paramètre	Type de données	Zone de mémoire	Description
—(MCR<)	Néant	–	–	Ouvre une zone MCR.

Relais de masquage hors fonction

L'opération « Relais de masquage hors fonction » (MCR>) ferme la zone MCR ouverte en dernier, en retirant de la pile MCR l'entrée RLG qui y avait été empilée par l'opération « Relais de masquage en fonction ». L'entrée libérée à l'autre extrémité de la pile MCR, qui fonctionne selon le principe « dernier entré, premier sorti », est mise à 1. Si la pile est déjà vide, l'opération « Relais de masquage hors fonction » provoque une erreur de pile MCR (MCRF).

Tableau 20-9 Relais de masquage hors fonction : représentation

Elément CONT	Paramètre	Type de données	Zone de mémoire	Description
—(MCR>)	Néant	–	–	Ferme la zone MCR ouverte en dernier.

Le relais de masquage est commandé par une pile MCR de 1 bit de large et pouvant contenir jusqu'à 8 entrées (voir figure 20-7). Le relais de masquage est activé tant que les huits entrées de la pile ne sont pas simultanément égales à 1. L'opération —(MCR<) copie le résultat logique (RLG) dans la pile MCR. L'opération —(MCR>) supprime la dernière entrée de la pile et met l'adresse libérée à 1. En cas d'erreur, lorsque plus de huit opérations —(MCR>) se suivent ou si vous tentez d'exécuter l'opération —(MCR>) alors que la pile est vide, par exemple, le programme affiche un message d'erreur MCRF. Le résultat du contrôle de la pile MCR figure après le pointeur de pile (MSP : 0 = vide, 1 = une entrée, 2 = deux entrées, ..., 8 = huit entrées).

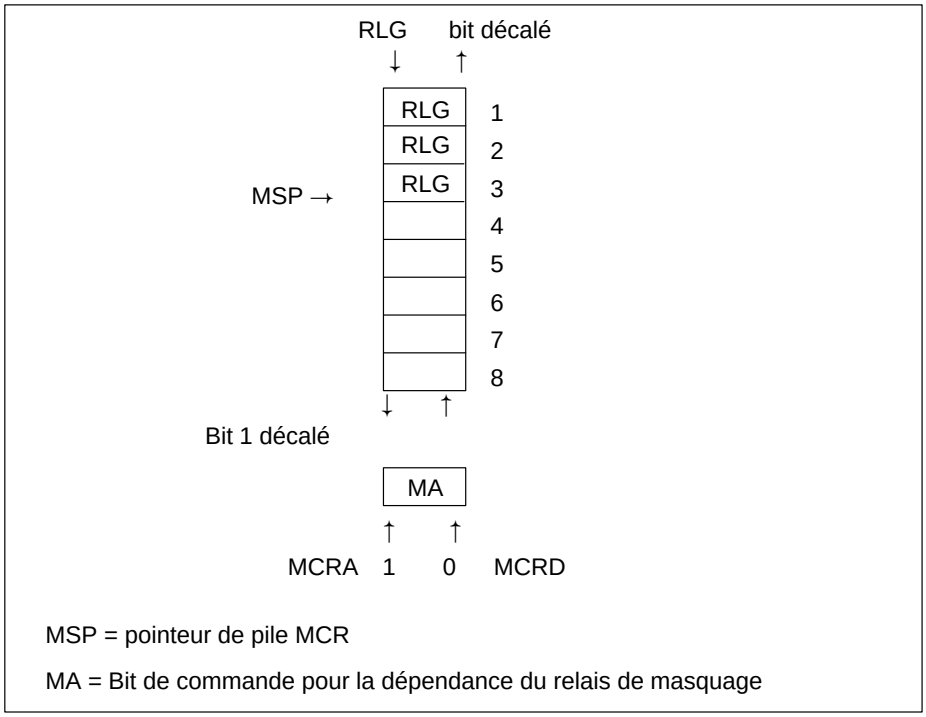


Figure 20-7 Pile du relais de masquage

Dans votre programme, vous ne devez jamais utiliser l'opération —(MCR<) sans l'opération —(MCR>).

L'opération —(MCR<) reprend l'état de signal du RLG pour le copier dans le bit MCR.

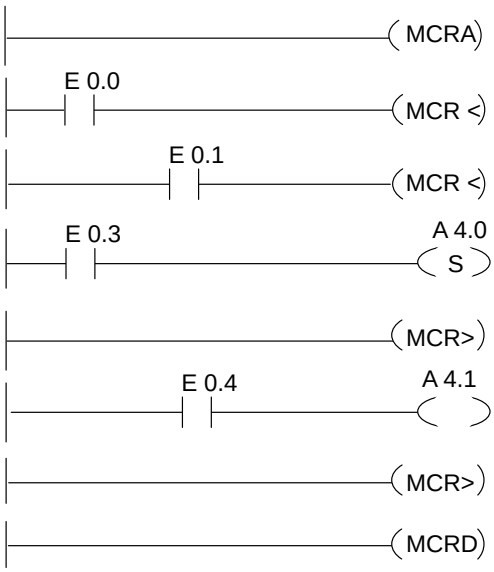
L'opération —(MCR>) met le bit MCR à 1 inconditionnellement. Pour cette raison, toute autre opération programmée entre les opérations —(MCRA) et —(MCRD) fonctionne indépendamment du bit MCR (voir figure ci-dessus).

Imbrication des opérations (MCR<) et (MCR>)

Vous pouvez imbriquer les opérations —(MCR<) et —(MCR>) en tenant compte de la limite maximale de niveaux d'imbrication (8). En d'autres termes, vous pouvez écrire jusqu'à huit opérations —(MCR<) consécutives avant d'insérer une opération —(MCR>). Dans votre programme, vous devez écrire le même nombre d'opérations —(MCR<) que d'opérations —(MCR>).

Si les opérations —(MCR<) sont imbriquées, le programme génère le bit MCR du niveau d'imbrication inférieur. Ensuite, l'opération —(MCR<) combine le RLG en cours au bit MCR en cours en fonction de la table de vérité ET.

Lorsqu'une opération —(MCR>) met fin à un niveau d'imbrication, elle extrait le bit MCR d'un niveau supérieur.



Lorsque l’opération —(MCRA) active la fonction MCR, vous pouvez créer jusqu’à huit zones MCR imbriquées. Dans notre exemple, il y en a deux. La première opération —(MCR>) va de pair avec la seconde opération —(MCR<). Toutes les opérations entre le deuxième jeu de crochets MCR (MCR<MCR>) appartiennent à la seconde zone MCR. Les opérations sont exécutées comme suit :

- Si l’état de signal de l’entrée E 0.0 est à 1, l’état de signal à l’entrée E 0.4 est affecté à la sortie A 4.1.
- Si l’état de signal de l’entrée E 0.0 est à 0, l’état de signal de la sortie A 4.1 est à 0 quel que soit l’état de signal à l’entrée E 0.4. La sortie A 4.0 reste inchangée quel que soit l’état de signal à l’entrée E 0.3.
- Si l’état de signal de l’entrée E 0.0 et de l’entrée E 0.1 est à 1, la sortie A 4.0 est mise à 1 si l’état de signal est 1 à l’entrée E 0.3 et que l’état de signal de la sortie A4.1 est égal à celui de l’entrée E 0.4.
- Si l’état de signal de l’entrée E 0.1 est à 0, la sortie A 4.0 reste inchangée quel que soit l’état de signal à l’entrée E 0.3 et à l’entrée E 0.0.

Description des bits du mot d’état

	RB	BI1	BI0	DEB	DM	OU	ETAT	RLG	/PI
écriture	–	–	–	–	–	0	1	–	0

Figure 20-8 Relais de masquage en fonction/hors fonction

Annexes

Liste alphabétique des opérations **A**

Exemples de programmation **B**

Représentation des nombres **C**

Bibliographie **D**

Liste alphabétique des opérations

A

Contenu de cette annexe

Paragraphe	Thème	Page
A.1	Liste des désignations françaises	A-2
A.2	Liste des désignations françaises et des désignations internationales (anglaises) correspondantes	A-5
A.3	Liste des désignations internationales (anglaises)	A-9
A.4	Liste des désignations internationales (anglaises) et des désignations françaises correspondantes	A-12

A.1 Liste des désignations françaises

Le tableau A-1 contient, par ordre alphabétique, la désignation française de chaque opération CONT, l'abréviation SIMATIC correspondante ainsi que la page où l'opération est expliquée.

Tableau A-1 Liste alphabétique des désignations françaises des opérations CONT avec les abréviations SIMATIC correspondantes

Désignation française	Abréviation SIMATIC	Page
Activer relais de masquage	—(MCRA)	20-9
Additionner entiers de 16 bits	ADD_I	11-2
Additionner entiers de 32 bits	ADD_DI	11-3
Additionner nombres réels	ADD_R	12-3
Affecter valeur	MOVE	14-2
Appeler FB sous forme de pavé	CALL_FB	20-4
Appeler FC sous forme de pavé	CALL_FC	20-4
Appeler FC/SFC sans paramètre	—(CALL)	20-2
Appeler SFB sous forme de pavé	CALL_SFB	20-4
Appeler SFC sous forme de pavé	CALL_SFC	20-4
Arrondir à entier de 32 bits	ROUND	14-15
Bascule mise à 0, mise à 1	RS	8-24
Bascule mise à 1, mise à 0	SR	8-23
Bit d'anomalie « Débordement »	OV —| |—	19-7
Bit d'anomalie « Débordement », forme inverse	OV —|/|—	19-7
Bit d'anomalie « Débordement mémorisé »	OS —| |—	19-9
Bit d'anomalie « Débordement mémorisé », forme inverse	OS —|/|—	19-9
Bit d'anomalie « Opération illicite »	UO —| |—	19-6
Bit d'anomalie « Opération illicite », forme inverse	UO —|/|—	19-6
Bit d'anomalie « Registre RB »	BIE —| |—	19-3
Bit d'anomalie « Registre RB », forme inverse	BIE —|/|—	19-3
Bit de résultat pour différent de 0	<>0 —| |—	19-4
Bit de résultat pour différent de 0, forme inverse	<>0 —|/|—	19-4
Bit de résultat pour égal à 0	==0 —| |—	19-4
Bit de résultat pour égal à 0, forme inverse	==0 —|/|—	19-4
Bit de résultat pour inférieur à 0	<0 —| |—	19-4
Bit de résultat pour inférieur à 0, forme inverse	<0 —|/|—	19-4
Bit de résultat pour inférieur ou égal à 0	<=0 —| |—	19-4
Bit de résultat pour inférieur ou égal à 0, forme inverse	<=0 —|/|—	19-4
Bit de résultat pour supérieur à 0	>0 —| |—	19-4
Bit de résultat pour supérieur à 0, forme inverse	>0 —|/|—	19-4
Bit de résultat pour supérieur ou égal à 0	>=0 —| |—	19-4
Bit de résultat pour supérieur ou égal à 0, forme inverse	>=0 —|/|—	19-4

Tableau A-1 Liste alphabétique des désignations françaises des opérations CONT avec les abréviations SIMATIC correspondantes (suite)

Désignation française	Abréviation SIMATIC	Page
Comparer entiers de 16 bits	CMP>=I	13-2
Comparer entiers de 32 bits	CMP>=D	13-3
Comparer nombres réels	CMP>=R	13-5
Complément à 1 d'entier de 16 bits	INV_I	14-10
Complément à 1 d'entier de 32 bits	INV_DI	14-11
Complément à 2 d'entier de 16 bits	NEG_I	14-12
Complément à 2 d'entier de 32 bits	NEG_DI	14-13
Compteur décrémental	Z_RUECK	10-7
Compteur incrémental	Z_VORW	10-5
Compteur incrémental/décrémental	ZAEHLER	10-3
Connecteur	——(#)——	8-6
Contact à fermeture	——| |——	8-3
Contact à ouverture	——|/|——	8-4
Convertir entier de 16 bits en entier de 32 bits	I_DI	14-6
Convertir entier de 16 bits en nombre DCB	I_BCD	14-5
Convertir entier de 32 bits en nombre réel	DI_R	14-9
Convertir entier de 32 bits en nombre DCB	DI_BCD	14-8
Convertir nombre réel en entier inférieur le plus proche	FLOOR	14-18
Convertir nombre réel en entier supérieur le plus proche	CEIL	14-17
Convertir nombre DCB en entier de 16 bits	BCD_I	14-4
Convertir nombre DCB en entier de 32 bits	BCD_DI	14-7
Décalage vers droite de double mot	SHR_DW	16-6
Décalage vers droite d'entier de 16 bits	SHR_I	16-7
Décalage vers droite d'entier de 32 bits	SHR_DI	16-9
Décalage vers droite de mot	SHR_W	16-6
Décalage vers gauche de double mot	SHL_DW	16-4
Décalage vers gauche de mot	SHL_W	16-2
Décrémenter	——(ZR)	8-13
Désactiver relais de masquage	——(MCRD)	20-9
Détecter front descendant de signal	NEG	8-22
Détecter front descendant du RLG	——(N)——	8-20
Détecter front montant de signal	POS	8-21
Détecter front montant du RLG	——(P)——	8-19
Diviser entiers de 16 bits	DIV_I	11-8
Diviser entiers de 32 bits	DIV_DI	11-9
Diviser nombres réels	DIV_R	12-6
ET double mot	WAND_DW	15-4
ET mot	WAND_W	15-3

Tableau A-1 Liste alphabétique des désignations françaises des opérations CONT avec les abréviations SIMATIC correspondantes (suite)

Désignation française	Abréviation SIMATIC	Page
Incrémenter	—(ZV)	8-12
Initialiser compteur	—(SZ)	8-11
Inverser RLG	—| NOT |—	8-7
Inverser le signe d'un nombre réel	NEG_R	14-14
Mettre à 0	—(R)	8-10
Mettre à 1	—(S)	8-9
Multiplier entiers de 16 bits	MUL_I	11-6
Multiplier entiers de 32 bits	MUL_DI	11-7
Multiplier nombres réels	MUL_R	12-5
OU double mot	WOR_DW	15-6
OU exclusif mot	WXOR_W	15-7
OU mot	WOR_W	15-5
OU exclusif double mot	WXOR_DW	15-8
Ouvrir bloc de données	—(AUF)	17-2
Relais de masquage en fonction	—(MCR<)	20-12
Relais de masquage hors fonction	—(MCR>)	20-12
Reste de division (32 bits)	MOD	11-10
Retour	—(RET)	20-7
Rotation vers droite de double mot	ROR_DW	16-12
Rotation vers gauche de double mot	ROL_DW	16-10
Saut si 1	—(JMP)	18-3
Saut si 0	—(JMPN)	18-5
Sauvegarder RLG dans RB	—(SAVE)	8-8
Sortie	—( )	8-5
Soustraire entiers de 16 bits	SUB_I	11-4
Soustraire entiers de 32 bits	SUB_DI	11-5
Soustraire nombres réels	SUB_R	12-4
Temporisation sous forme d'impulsion	S_IMPULS	9-5
Temporisation sous forme d'impulsion	—(SI)	8-14
Temporisation sous forme d'impulsion prolongée	S_VIMP	9-7
Temporisation sous forme d'impulsion prolongée	—(SV)	8-15
Temporisation sous forme de retard à la montée	S_EVERZ	9-9
Temporisation sous forme de retard à la montée	—(SE)	8-16
Temporisation sous forme de retard à la montée mémorisé	S_SEVERZ	9-11
Temporisation sous forme de retard à la montée mémorisé	—(SS)	8-17
Temporisation sous forme de retard à la retombée	S_AVERZ	9-13
Temporisation sous forme de retard à la retombée	—(SA)	8-18
Tronquer à la partie entière (32 bits)	TRUNC	14-16

A.2 Liste des désignations françaises et des désignations internationales (anglaises) correspondantes

Le tableau A-2 contient, par ordre alphabétique, la désignation française de chaque opération CONT, la désignation internationale (anglaise) correspondante ainsi que la page où l'opération est expliquée.

Tableau A-2 Liste alphabétique des désignations françaises des opérations CONT avec les désignations internationales (anglaises) correspondantes

Désignation française	Désignation internationale (anglaise)	Page
Activer relais de masquage	Master Control Relay Activate	20-9
Additionner entiers de 16 bits	Add Integer	11-2
Additionner entiers de 32 bits	Add Double Integer	11-3
Additionner nombres réels	Add Real	12-3
Affecter valeur	Assign a Value	14-2
Appeler FB sous forme de pavé	Call FB from Box	20-4
Appeler FC sous forme de pavé	Call FC from Box	20-4
Appeler FC/SFC sans paramètre	Call FC SFC from Coil (without parameters)	20-2
Appeler SFB sous forme de pavé	Call System FB from Box	20-4
Appeler SFC sous forme de pavé	Call System FC from Box	20-4
Arrondir à entier de 32 bits	Round to Double Integer	14-15
Bascule mise à 0, mise à 1	Reset_Set Flip Flop	8-24
Bascule mise à 1, mise à 0	Set_Reset Flip Flop	8-23
Bit d'anomalie « Débordement »	Exception Bit Overflow	19-7
Bit d'anomalie « Débordement », forme inverse	Negated Exception Bit Overflow	19-7
Bit d'anomalie « Débordement mémorisé »	Exception Bit Overflow Stored	19-9
Bit d'anomalie « Débordement mémorisé », forme inverse	Negated Exception Bit Overflow Stored	19-9
Bit d'anomalie « Opération illicite »	Exception Bit Unordered	19-6
Bit d'anomalie « Opération illicite », forme inverse	Negated Exception Bit Unordered	19-6
Bit d'anomalie « Registre RB »	Exception Bit BR Memory	19-3
Bit d'anomalie « Registre RB », forme inverse	Negated Exception Bit BR Memory	19-3
Bit de résultat pour différent de 0	Result Bit Not Equal 0	19-4
Bit de résultat pour différent de 0, forme inverse	Negated Result Bit Not Equal 0	19-4
Bit de résultat pour égal à 0	Result Bit Equal 0	19-4
Bit de résultat pour égal à 0, forme inverse	Negated Result Bit Equal 0	19-4
Bit de résultat pour inférieur à 0	Result Bit Less Than 0	19-4
Bit de résultat pour inférieur à 0, forme inverse	Negated Result Bit Less Than 0	19-4
Bit de résultat pour inférieur ou égal à 0	Result Bit Less Equal 0	19-4
Bit de résultat pour inférieur ou égal à 0, forme inverse	Negated Result Bit Less Equal 0	19-4
Bit de résultat pour supérieur à 0	Result Bit Greater Than 0	19-4

Tableau A-2 Liste alphabétique des désignations françaises des opérations CONT avec les désignations internationales (anglaises) correspondantes (suite)

Désignation française	Désignation internationale (anglaise)	Page
Bit de résultat pour supérieur à 0, forme inverse	Negated Result Bit Greater Than 0	19-4
Bit de résultat pour supérieur ou égal à 0	Result Bit Greater Equal 0	19-4
Bit de résultat pour supérieur ou égal à 0, forme inverse	Negated Result Bit Greater Equal 0	19-4
Comparer entiers de 16 bits	Compare Integer (>, <, ==, <>, <=, >=)	13-2
Comparer entiers de 32 bits	Compare Double Integer (>, <, ==, <>, <=, >=)	13-3
Comparer nombres réels	Compare Real (>, <, ==, <>, <=, >=)	13-5
Complément à 1 d'entier de 16 bits	ONEs Complement Integer	14-10
Complément à 1 d'entier de 32 bits	ONEs Complement Double Integer	14-11
Complément à 2 d'entier de 16 bits	TWOs Complement Integer	14-12
Complément à 2 d'entier de 32 bits	TWOs Complement Double Integer	14-13
Compteur décrémental	Down Counter	10-7
Compteur incrémental	Up Counter	10-5
Compteur incrémental/décrémental	Up-Down Counter	10-3
Connecteur	Midline Output	8-6
Contact à fermeture	Normally Open Contact (Address)	8-3
Contact à ouverture	Normally Closed Contact (Address)	8-4
Convertir entier de 16 bits en entier de 32 bits	Integer to Double Integer	14-6
Convertir entier de 16 bits en nombre DCB	Integer to BCD	14-5
Convertir entier de 32 bits en nombre réel	Double Integer to Real	14-9
Convertir entier de 32 bits en nombre DCB	Double Integer to BCD	14-8
Convertir nombre réel en entier inférieur le plus proche	Floor	14-18
Convertir nombre réel en entier supérieur le plus proche	Ceiling	14-17
Convertir nombre DCB en entier de 16 bits	BCD to Integer	14-4
Convertir nombre DCB en entier de 32 bits	BCD to Double Integer	14-7
Décalage vers droite de double mot	Shift Right Double Word	16-6
Décalage vers droite d'entier de 16 bits	Shift Right Integer	16-7
Décalage vers droite d'entier de 32 bits	Shift Right Double Integer	16-9
Décalage vers droite de mot	Shift Right Word	16-6
Décalage vers gauche de double mot	Shift Left Double Word	16-4
Décalage vers gauche de mot	Shift Left Word	16-2
Décrémenter	Down Counter Coil	8-13
Désactiver relais de masquage	Master Control Relay Deactivate	20-9
Détecter front descendant de signal	Address Negative Edge Detection	8-22
Détecter front descendant du RLG	Negative RLO Edge Detection	8-20
Détecter front montant de signal	Address Positive Edge Detection	8-21
Détecter front montant du RLG	Positive RLO Edge Detection	8-19

Tableau A-2 Liste alphabétique des désignations françaises des opérations CONT avec les désignations internationales (anglaises) correspondantes (suite)

Désignation française	Désignation internationale (anglaise)	Page
Diviser entiers de 16 bits	Divide Integer	11-8
Diviser entiers de 32 bits	Divide Double Integer	11-9
Diviser nombres réels	Divide Real	12-6
ET double mot	(Word) And Double Word	15-4
ET mot	(Word) And Word	15-3
Incrémenter	Up Counter Coil	8-12
Initialiser compteur	Set Counter Value	8-11
Inverser RLG	Invert Power Flow	8-7
Inverser le signe d'un nombre réel	Negate Real Number	14-14
Mettre à 0	Reset Coil	8-10
Mettre à 1	Set Coil	8-9
Multiplier entiers de 16 bits	Multiply Integer	11-6
Multiplier entiers de 32 bits	Multiply Double Integer	11-7
Multiplier nombres réels	Multiply Real	12-5
OU double mot	(Word) Or Double Word	15-6
OU exclusif double mot	(Word) Exclusive Or Double Word	15-8
OU exclusif mot	(Word) Exclusive Or Word	15-7
OU mot	(Word) Or Word	15-5
Ouvrir bloc de données	Open Data Block: DB or DI	17-2
Relais de masquage en fonction	Master Control Relay On	20-12
Relais de masquage hors fonction	Master Control Relay Off	20-12
Reste de division (32 bits)	Return Fraction Double Integer	11-10
Retour	Return	20-7
Rotation vers droite de double mot	Rotate Right Double Word	16-12
Rotation vers gauche de double mot	Rotate Left Double Word	16-10
Saut si 1	Jump	18-3
Saut si 0	Jump-If-Not	18-5
Sauvegarder RLG dans RB	Save RLO to BR Memory	8-8
Sortie	Output Coil	8-5
Soustraire entiers de 16 bits	Subtract Integer	11-4
Soustraire entiers de 32 bits	Subtract Double Integer	11-5
Soustraire nombres réels	Subtract Real	12-4
Temporisation sous forme d'impulsion	Pulse S5 Timer	9-5
Temporisation sous forme d'impulsion	Pulse Timer Coil	8-14

Tableau A-2 Liste alphabétique des désignations françaises des opérations CONT avec les désignations internationales (anglaises) correspondantes (suite)

Désignation française	Désignation internationale (anglaise)	Page
Temporisation sous forme d'impulsion prolongée	Extended Pulse S5 Timer	9-7
Temporisation sous forme d'impulsion prolongée	Extended Pulse Timer Coil	8-15
Temporisation sous forme de retard à la montée	On-Delay S5 Timer	9-9
Temporisation sous forme de retard à la montée	On-Delay Timer Coil	8-16
Temporisation sous forme de retard à la montée mémorisé	Retentive On-Delay S5 Timer	9-11
Temporisation sous forme de retard à la montée mémorisé	Retentive On-Delay Timer Coil	8-17
Temporisation sous forme de retard à la retombée	Off-Delay S5 Timer	9-13
Temporisation sous forme de retard à la retombée	Off-Delay Timer Coil	8-18
Tronquer à la partie entière (32 bits)	Truncate Double Integer Part	14-16

A.3 Liste des désignations internationales (anglaises)

Le tableau A-3 contient, par ordre alphabétique, la désignation internationale (anglaise) de chaque opération CONT, l'abréviation internationale correspondante ainsi que la page où l'opération est expliquée.

Tableau A-3 Liste alphabétique des désignations internationales (anglaises) des opérations CONT avec les abréviations internationales correspondantes

Désignation internationale (anglaise)	Abréviation internationale	Page
Add Double Integer	ADD_DI	11-3
Add Integer	ADD_I	11-2
Add Real	ADD_R	12-3
Address Negative Edge Detection	NEG	8-22
Address Positive Edge Detection	POS	8-21
Assign a Value	MOVE	14-2
BCD to Double Integer	BCD_DI	14-7
BCD to Integer	BCD_I	14-4
Call FB from Box	CALL_FB	20-4
Call FC from Box	CALL_FC	20-4
Call FC SFC from Coil (without parameters)	——(CALL)	20-2
Call System FB from Box	CALL_SFB	20-4
Call System FC from Box	CALL_SFC	20-4
Ceiling	CEIL	14-17
Compare Double Integer (>, <, ==, <>, <=, >=)	CMP>=D	13-3
Compare Integer (>, <, ==, <>, <=, >=)	CMP>=I	13-2
Compare Real (>, <, ==, <>, <=, >=)	CMP>=R	13-5
Divide Double Integer	DIV_DI	11-9
Divide Integer	DIV_I	11-8
Divide Real	DIV_R	12-6
Double Integer to BCD	DI_BCD	14-8
Double Integer to Real	DI_R	14-9
Down Counter	S_CD	10-7
Down Counter Coil	——(CD)	8-13
Exception Bit BR Memory	BR —| |—	19-3
Exception Bit Overflow	OV —| |—	19-7
Exception Bit Overflow Stored	OS —| |—	19-9
Exception Bit Unordered	UO —| |—	19-6
Extended Pulse S5 Timer	S_PEXT	9-7
Extended Pulse Timer Coil	—(SE)	8-15
Floor	FLOOR	14-18
Integer to BCD	I_BCD	14-5
Integer to Double Integer	I_DI	14-6

Tableau A-3 Liste alphabétique des désignations internationales (anglaises) des opérations CONT avec les abréviations internationales correspondantes (suite)

Désignation internationale (anglaise)	Abréviation internationale	Page
Invert Power Flow	—| NOT |—	8-7
Jump	—(JMP)	18-3
Jump-If-Not	—(JMPN)	18-5
Master Control Relay Activate	—(MCRA)	20-9
Master Control Relay Deactivate	—(MCRD)	20-9
Master Control Relay Off	—(MCR>)	20-12
Master Control Relay On	—(MCR<)	20-12
Midline Output	—(#)—	8-6
Multiply Double Integer	MUL_DI	11-7
Multiply Integer	MUL_I	11-6
Multiply Real	MUL_R	12-5
Negated Exception Bit BR Memory	BR —|/|—	19-3
Negated Exception Bit Overflow	OV —|/|—	19-7
Negated Exception Bit Overflow Stored	OS —|/|—	19-9
Negated Exception Bit Unordered	UO —|/|—	19-6
Negated Result Bit Equal 0	==0 —|/|—	19-4
Negated Result Bit Greater Equal 0	>=0 —|/|—	19-4
Negated Result Bit Greater Than 0	>0 —|/|—	19-4
Negated Result Bit Less Equal 0	<=0 —|/|—	19-4
Negated Result Bit Less Than 0	<0 —|/|—	19-4
Negated Result Bit Not Equal 0	<>0 —|/|—	19-4
Negative RLO Edge Detection	—(N)—	8-20
Negate Real Number	NEG_R	14-14
Normally Closed Contact (Address)	—|/|—	8-4
Normally Open Contact (Address)	—| |—	8-3
Off-Delay S5 Timer	S_OFFDT	9-13
Off-Delay Timer Coil	—(SF)	8-18
On-Delay S5 Timer	S_ODT	9-9
On-Delay Timer Coil	—(SD)	8-16
ONEs Complement Double Integer	INV_DI	14-11
ONEs Complement Integer	INV_I	14-10
Open Data Block: DB or DI	—(OPN)	17-2
Output Coil	—( )	8-5
Positive RLO Edge Detection	—(P)—	8-19
Pulse S5 Timer	S_PULSE	9-5
Pulse Timer Coil	—(SP)	8-14
Reset Coil	—(R)	8-10
Reset_Set Flip Flop	RS	8-24

Tableau A-3 Liste alphabétique des désignations internationales (anglaises) des opérations CONT avec les abréviations internationales correspondantes (suite)

Désignation internationale (anglaise)	Abréviation internationale	Page
Result Bit Equal 0	==0 —| |—	1-6
Result Bit Greater Equal 0	>=0 —| |—	19-4
Result Bit Greater Than 0	>0 —| |—	19-4
Result Bit Less Equal 0	<=0 —| |—	19-4
Result Bit Less Than 0	<0 —| |—	19-4
Result Bit Not Equal 0	<>0 —| |—	19-4
Retentive On-Delay S5 Timer	S_ODTS	9-11
Retentive On-Delay Timer Coil	—(SS)	8-17
Return	—(RET)	20-7
Return Fraction Double Integer	MOD	11-10
Rotate Left Double Word	ROL_DW	16-10
Rotate Right Double Word	ROR_DW	16-12
Round to Double Integer	ROUND	14-15
Save RLO to BR Memory	—(SAVE)	8-8
Set Coil	—(S)	8-9
Set Counter Value	—(SC)	8-11
Set_Reset Flip Flop	SR	8-23
Shift Left Double Word	SHL_DW	16-4
Shift Left Word	SHL_W	16-2
Shift Right Double Integer	SHR_DI	16-9
Shift Right Double Word	SHR_DW	16-6
Shift Right Integer	SHR_I	16-7
Shift Right Word	SHR_W	16-6
Subtract Double Integer	SUB_DI	11-5
Subtract Integer	SUB_I	11-4
Subtract Real	SUB_R	12-4
Truncate Double Integer Part	TRUNC	14-16
TWOs Complement Double Integer	NEG_DI	14-13
TWOs Complement Integer	NEG_I	14-12
Up Counter	S_CU	10-5
Up Counter Coil	—(CU)	8-12
Up–Down Counter	S_CUD	10-3
(Word) And Double Word	WAND_DW	15-4
(Word) And Word	WAND_W	15-3
(Word) Exclusive Or Double Word	WXOR_DW	15-8
(Word) Exclusive Or Word	WXOR_W	15-7
(Word) Or Double Word	WOR_DW	15-6
(Word) Or Word	WOR_W	15-5

A.4 Liste des désignations internationales (anglaises) et des désignations françaises correspondantes

Le tableau A-4 contient, par ordre alphabétique, la désignation internationale (anglaise) de chaque opération CONT, la désignation française correspondante ainsi que la page où l'opération est expliquée.

Tableau A-4 Liste alphabétique des désignations internationales (anglaises) des opérations CONT avec les désignations françaises correspondantes

Désignation internationale (anglaise)	Désignation française correspondante	Page
Add Double Integer	Additionner entiers de 32 bits	11-3
Add Integer	Additionner entiers de 16 bits	11-2
Add Real	Additionner nombres réels	12-3
Address Negative Edge Detection	Détecter front descendant de signal	8-22
Address Positive Edge Detection	Détecter front montant de signal	8-21
Assign a Value	Affecter valeur	14-2
BCD to Double Integer	Convertir nombre DCB en entier de 32 bits	14-7
BCD to Integer	Convertir nombre DCB en entier de 16 bits	14-4
Call FB from Box	Appeler FB sous forme de pavé	20-4
Call FC from Box	Appeler FC sous forme de pavé	20-4
Call FC SFC from Coil (without parameters)	Appeler FC/SFC sans paramètre	20-2
Call System FB from Box	Appeler SFB sous forme de pavé	20-4
Call System FC from Box	Appeler SFC sous forme de pavé	20-4
Ceiling	Convertir nombre réel en entier supérieur le plus proche	14-17
Compare Double Integer (>, <, ==, <>, <=, >=)	Comparer entiers de 32 bits	13-3
Compare Integer (>, <, ==, <>, <=, >=)	Comparer entiers de 16 bits	13-2
Compare Real (>, <, ==, <>, <=, >=)	Comparer nombres réels	13-5
Divide Double Integer	Diviser entiers de 32 bits	11-9
Divide Integer	Diviser entiers de 16 bits	11-8
Divide Real	Diviser nombres réels	12-6
Double Integer to BCD	Convertir entier de 32 bits en nombre DCB	14-8
Double Integer to Real	Convertir entier de 32 bits en nombre réel	14-9
Down Counter	Compteur décrémental	10-7
Down Counter Coil	Décrémenter	8-13
Exception Bit BR Memory	Bit d'anomalie « Registre RB »	19-3
Exception Bit Overflow	Bit d'anomalie « Débordement »	19-7
Exception Bit Overflow Stored	Bit d'anomalie « Débordement mémorisé »	19-9
Exception Bit Unordered	Bit d'anomalie « Opération illicite »	19-6
Extended Pulse S5 Timer	Temporisation sous forme d'impulsion prolongée	9-7
Extended Pulse Timer Coil	Temporisation sous forme d'impulsion prolongée	8-15

Tableau A-4 Liste alphabétique des désignations internationales (anglaises) des opérations CONT avec les désignations françaises correspondantes (suite)

Désignation internationale (anglaise)	Désignation française correspondante	Page
Floor	Convertir nombre réel en entier inférieur le plus proche	14-18
Integer to BCD	Convertir entier de 16 bits en nombre DCB	14-5
Integer to Double Integer	Convertir entier de 16 bits en entier de 32 bits	14-6
Invert Power Flow	Inverser RLG	8-7
Jump	Saut si 1	18-3
Jump-If-Not	Saut si 0	18-5
Master Control Relay Activate	Activer relais de masquage	20-9
Master Control Relay Deactivate	Désactiver relais de masquage	20-9
Master Control Relay Off	Relais de masquage hors fonction	20-12
Master Control Relay On	Relais de masquage en fonction	20-12
Midline Output	Connecteur	8-6
Multiply Double Integer	Multiplier entiers de 32 bits	11-7
Multiply Integer	Multiplier entiers de 16 bits	11-6
Multiply Real	Multiplier nombres réels	12-5
Negated Exception Bit BR Memory	Bit d'anomalie « Registre RB », forme inverse	19-3
Negated Exception Bit Overflow	Bit d'anomalie « Débordement », forme inverse	19-7
Negated Exception Bit Overflow Stored	Bit d'anomalie « Débordement mémorisé », forme inverse	19-9
Negated Exception Bit Unordered	Bit d'anomalie « Opération illicite », forme inverse	19-6
Negated Result Bit Equal 0	Bit de résultat pour égal à 0, forme inverse	19-4
Negated Result Bit Greater Equal 0	Bit de résultat pour supérieur ou égal à 0, forme inverse	19-4
Negated Result Bit Greater Than 0	Bit de résultat pour supérieur à 0, forme inverse	19-4
Negated Result Bit Less Equal 0	Bit de résultat pour inférieur ou égal à 0, forme inverse	19-4
Negated Result Bit Less Than 0	Bit de résultat pour inférieur à 0, forme inverse	19-4
Negated Result Bit Not Equal 0	Bit de résultat pour différent de 0, forme inverse	19-4
Negative RLO Edge Detection	Détecter front descendant du RLG	8-20
Negate Real Number	Inverser le signe d'un nombre réel	14-14
Normally Closed Contact (Address)	Contact à ouverture	8-4
Normally Open Contact (Address)	Contact à fermeture	8-3
Off-Delay S5 Timer	Temporisation sous forme de retard à la retombée	9-13
Off-Delay Timer Coil	Temporisation sous forme de retard à la retombée	8-18
On-Delay S5 Timer	Temporisation sous forme de retard à la montée	9-9
On-Delay Timer Coil	Temporisation sous forme de retard à la montée	8-16
ONEs Complement Double Integer	Complément à 1 d'entier de 32 bits	14-11
ONEs Complement Integer	Complément à 1 d'entier de 16 bits	14-10
Open Data Block: DB or DI	Ouvrir bloc de données	17-2
Output Coil	Sortie	8-5
Positive RLO Edge Detection	Détecter front montant du RLG	8-19

Tableau A-4 Liste alphabétique des désignations internationales (anglaises) des opérations CONT avec les désignations françaises correspondantes (suite)

Désignation internationale (anglaise)	Désignation française correspondante	Page
Pulse S5 Timer	Temporisation sous forme d'impulsion	9-5
Pulse Timer Coil	Temporisation sous forme d'impulsion	8-14
Reset Coil	Mettre à 0	8-10
Reset_Set Flip Flop	Bascule mise à 0, mise à 1	8-24
Result Bit Equal 0	Bit de résultat pour égal à 0	19-4
Result Bit Greater Equal 0	Bit de résultat pour supérieur ou égal à 0	19-4
Result Bit Greater Than 0	Bit de résultat pour supérieur à 0	19-4
Result Bit Less Equal 0	Bit de résultat pour inférieur ou égal à 0	19-4
Result Bit Less Than 0	Bit de résultat pour inférieur à 0	19-4
Result Bit Not Equal 0	Bit de résultat pour différent de 0	19-4
Retentive On-Delay S5 Timer	Temporisation sous forme de retard à la montée mémorisé	9-11
Retentive On-Delay Timer Coil	Temporisation sous forme de retard à la montée mémorisé	8-17
Return	Retour	20-7
Return Fraction Double Integer	Reste de division (32 bits)	11-10
Rotate Left Double Word	Rotation vers gauche de double mot	16-10
Rotate Right Double Word	Rotation vers droite de double mot	16-12
Round to Double Integer	Arrondir à entier de 32 bits	14-15
Save RLO to BR Memory	Sauvegarder RLG dans RB	8-8
Set Coil	Mettre à 1	8-9
Set Counter Value	Initialiser compteur	8-11
Set_Reset Flip Flop	Bascule mise à 1, mise à 0	8-23
Shift Left Double Word	Décalage vers gauche de double mot	16-4
Shift Left Word	Décalage vers gauche de mot	16-2
Shift Right Double Integer	Décalage vers droite d'entier de 32 bits	16-9
Shift Right Double Word	Décalage vers droite de double mot	16-6
Shift Right Integer	Décalage vers droite d'entier de 16 bits	16-7
Shift Right Word	Décalage vers droite de mot	16-6
Subtract Double Integer	Soustraire entiers de 32 bits	11-5
Subtract Integer	Soustraire entiers de 16 bits	11-4
Subtract Real	Soustraire nombres réels	12-4
Truncate Double Integer Part	Tronquer à la partie entière (32 bits)	14-16
TWOs Complement Double Integer	Complément à 2 d'entier de 32 bits	14-13
TWOs Complement Integer	Complément à 2 d'entier de 16 bits	14-12
Up Counter	Compteur incrémental	10-5
Up Counter Coil	Incrémenter	8-12
Up-Down Counter	Compteur incrémental/décrémental	10-3
(Word) And Double Word	ET double mot	15-4

Tableau A-4 Liste alphabétique des désignations internationales (anglaises) des opérations CONT avec les désignations françaises correspondantes (suite)

Désignation internationale (anglaise)	Désignation française correspondante	Page
(Word) And Word	ET mot	15-3
(Word) Exclusive Or Double Word	OU exclusif double mot	15-8
(Word) Exclusive Or Word	OU exclusif mot	15-7
(Word) Or Double Word	OU double mot	15-6
(Word) Or Word	OU mot	15-5

Tableau A-5 Liste des opérations CONT mentionnées dans ce manuel avec les abréviations SIMATIC et les abréviations internationales

Désignation française	Abréviation SIMATIC	Abréviation internationale	Page
Bit d'anomalie « Registre RB »	BIE —| |—	BR —| |—	19-3
Compteur décrémental	Z_RUECK	S_CD	10-7
Compteur incrémental	Z_VORW	S_CU	10-5
Compteur incrémental/décrémental	ZAEHLER	S_CUD	10-3
Décrémenter	——(ZR)	——(CD)	8-13
Incrémenter	—(ZV)	—(CU)	8-12
Initialiser compteur	—(SZ)	—(SC)	8-11
Ouvrir bloc de données	—(AUF)	—(OPN)	17-2
Temporisation sous forme d'impulsion	S_IMPULS	S_PULSE	9-5
Temporisation sous forme d'impulsion	—(SI)	—(SP)	8-14
Temporisation sous forme d'impulsion prolongée	S_VIMP	S_PEXT	9-7
Temporisation sous forme d'impulsion prolongée	—(SV)	—(SE)	8-15
Temporisation sous forme de retard à la montée	S_EVERZ	S_ODT	9-9
Temporisation sous forme de retard à la montée	—(SE)	—(SD)	8-16
Temporisation sous forme de retard à la montée mémorisé	S_SEVERZ	S_ODTS	9-11
Temporisation sous forme de retard à la montée mémorisé	—(SS)	—(SS)	8-17
Temporisation sous forme de retard à la retombée	S_AVERZ	S_OFFDT	9-13
Temporisation sous forme de retard à la retombée	—(SA)	—(SF)	8-18

Exemples de programmation

B

Contenu de cette annexe

Paragraphe	Thème	Page
B.1	Présentation	B-2
B.2	Opérations de combinaison sur bits	B-3
B.3	Opérations de temporisation	B-7
B.4	Opérations de comptage et de comparaison	B-11
B.5	Opérations arithmétiques sur nombres entiers	B-13
B.6	Opérations de combinaison sur mots	B-14

B.1 Présentation

Applications pratiques

Chacune des opérations CONT décrites dans ce manuel déclenche une fonction précise. En combinant ces opérations dans un programme, vous pouvez exécuter une grande variété de tâches d'automatisation. Vous trouverez dans la suite de ce chapitre quelques exemples d'applications pratiques des opérations CONT :

- Commande d'un tapis roulant à l'aide d'opérations de combinaison sur bits
- Détection du sens de déplacement d'un tapis roulant à l'aide d'opérations de combinaison sur bits
- Génération d'une période d'horloge à l'aide d'opérations de temporisation
- Surveillance de l'espace de stockage à l'aide d'opérations de comptage et de comparaison
- Calculs à l'aide d'opérations arithmétiques sur nombres entiers
- Réglage de la durée de chauffage d'un four

Opérations utilisées

Les exemples dans cette annexe font appel aux opérations suivantes :

- Additionner entiers de 16 bits (ADD_I)
- Affecter valeur (MOVE)
- Comparer entiers de 16 bits (CMP_I<=)
- Comparer entiers de 16 bits (CMP_I>=)
- Contact à fermeture —| |—
- Contact à ouverture —|/|—
- Décrémenter —(Z_RUECK)
- Détecter front montant du RLG —(P)—
- Diviser entiers de 16 bits (DIV_I)
- ET mot (WAND_W)
- Incrémenter —(Z_VORW)
- Mettre à 0 —(R)
- Mettre à 1 —(S)
- Multiplier entiers de 16 bits (MUL_I)
- Retour —(RET)
- Saut si 0 (JMPN)—
- Sortie —( )
- Temporisation sous forme d'impulsion prolongée —(SV)—
- OU mot (WOR_W)

B.2 Opérations combinatoires sur bits

Commande d'un tapis roulant

La figure B-1 montre un tapis roulant pouvant être mis en route électriquement. Deux boutons-poussoirs, S1 pour MARCHE et S2 pour ARRET, se situent au début du tapis et deux, S3 pour MARCHE et S4 pour ARRET, à la fin du tapis. Il est donc possible de démarrer et d'arrêter le tapis à ses deux extrémités. D'autre part, le capteur S5 arrête le tapis lorsqu'un objet atteint la fin du tapis.

Programmation symbolique

Vous pouvez écrire le programme de commande du tapis roulant présenté à la figure B-1 en représentant les diverses composantes du système convoyeur à l'aide de mnémoniques. Si vous choisissez cette méthode, vous devez créer une table des mnémoniques afin de mettre les mnémoniques choisis en relation avec les adresses absolues (voir tableau B-1). Vous définissez les mnémoniques dans la table des mnémoniques (voir le *Guide de l'utilisateur* /**231**/).

Tableau B-1 Eléments de programmation symbolique pour un système convoyeur

Composante du système	Désignation absolue	Mnémonique	Table des mnémoniques
Bouton-poussoir Marche	E 1.1	S1	E 1.1 S1
Bouton-poussoir Arrêt	E 1.2	S2	E 1.2 S2
Bouton-poussoir Marche	E 1.3	S3	E 1.3 S3
Bouton-poussoir Arrêt	E 1.4	S4	E 1.4 S4
Capteur	E 1.5	S5	E 1.5 S5
Moteur	A 4.0	MOTEUR_MAR	A 4.0 MOTEUR_MAR

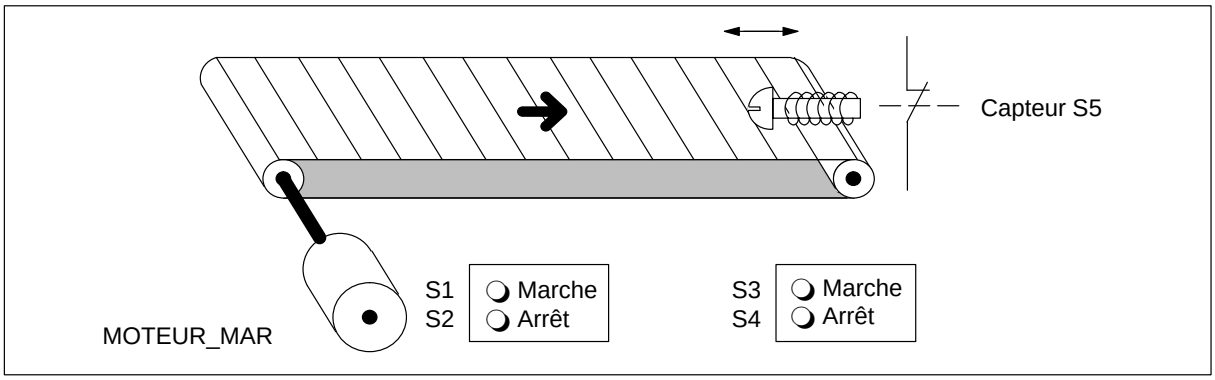


Figure B-1 Système convoyeur

Programmation absolue

Vous pouvez écrire le programme de commande du tapis roulant présenté à la figure B-1 en représentant les diverses composantes du système convoyeur à l'aide d'adresses absolues (voir tableau B-2). La figure B-2 illustre un programme CONT pour commander un tapis roulant.

Tableau B-2 Eléments de programmation absolue pour un système convoyeur

Composante du système	Désignation absolue
Bouton-poussoir Marche	E 1.1
Bouton-poussoir Arrêt	E 1.2
Bouton-poussoir Marche	E 1.3
Bouton-poussoir Arrêt	E 1.4
Capteur	E 1.5
Moteur	A 4.0

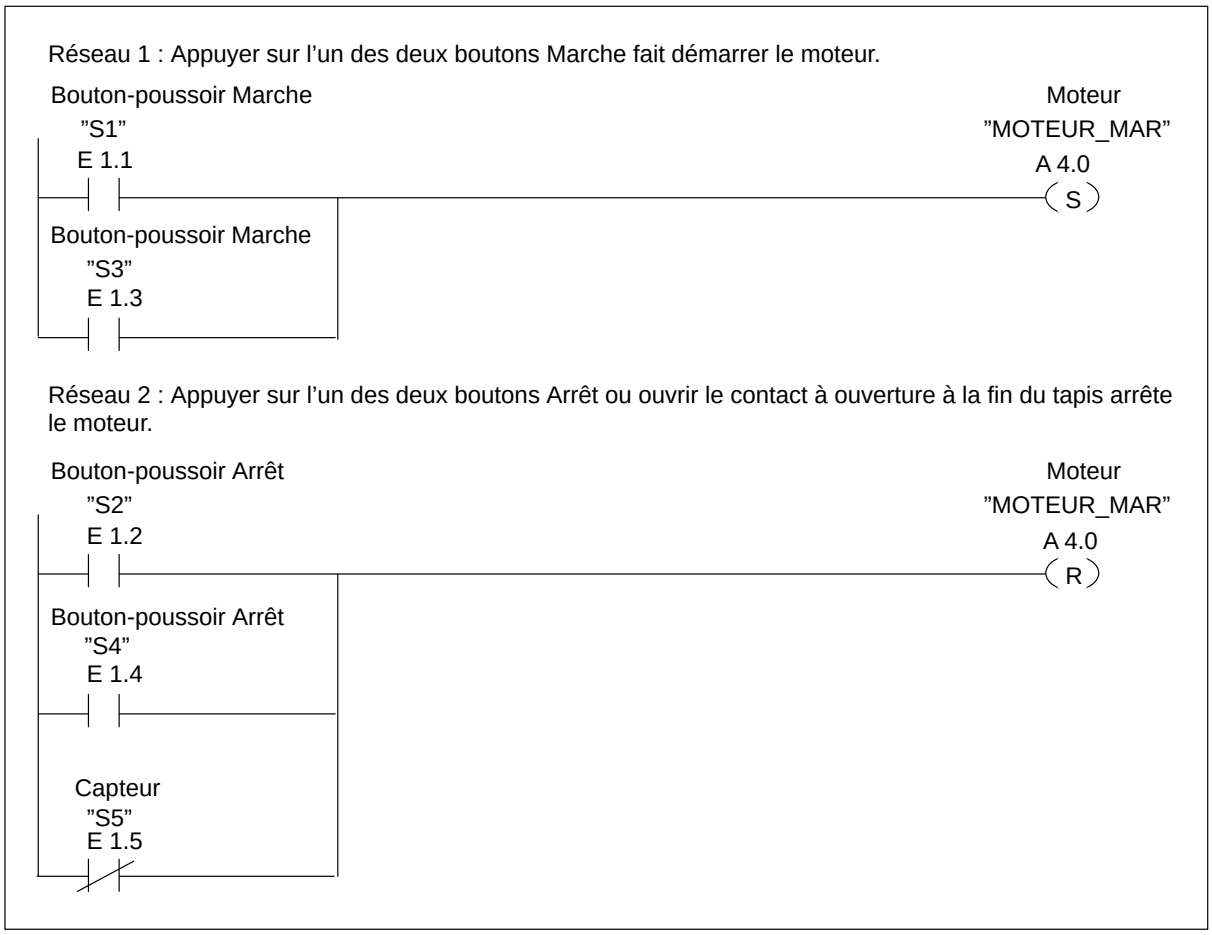


Figure B-2 Schéma à contacts pour commander un tapis roulant

Détection du sens de déplacement d'un tapis roulant

La figure B-3 montre un tapis roulant équipé de deux barrières photoélectriques (BPE1 et BPE2) chargées de détecter le sens dans lequel se déplace un paquet sur le tapis. Chaque barrière photoélectrique fonctionne comme un contact à fermeture (voir paragraphe 8.2).

Programmation symbolique

Vous pouvez écrire un programme qui active l'affichage du sens de déplacement du tapis roulant présenté à la figure B-3 en représentant, à l'aide de mnémoniques, les diverses composantes du système convoyeur, y compris les barrières photoélectriques qui détectent le sens de déplacement. Si vous choisissez cette méthode, vous devez créer une table des mnémoniques afin de mettre les mnémoniques choisis en relation avec les adresses absolues (voir tableau B-3). Vous définissez les mnémoniques dans la table des mnémoniques (voir le *Guide de l'utilisateur* /**231**/).

Tableau B-3 Eléments de programmation symbolique pour la détection du sens de déplacement

Composante du système	Désignation absolue	Mnémonique	Table des mnémoniques
Barrière photoélectrique 1	E 0.0	BPE1	E 0.0 BPE1
Barrière photoélectrique 2	E 0.1	BPE2	E 0.1 BPE2
Affichage pour mouvement vers la droite	A 4.0	DROITE	A 4.0 DROITE
Affichage pour mouvement vers la gauche	A 4.1	GAUCHE	A 4.1 GAUCHE
Mémento de cadence 1	M 0.0	MP1	M 0.0 MP1
Mémento de cadence 2	M 0.1	MP2	M 0.1 MP2

Programmation absolue

Vous pouvez écrire un programme qui active l'affichage du sens de déplacement du tapis roulant présenté à la figure B-3 en représentant, à l'aide d'adresses absolues, les barrières photoélectriques qui détectent le sens de déplacement (voir tableau B-4). La figure B-3 montre un programme CONT pour activer l'affichage du sens de déplacement du tapis roulant.

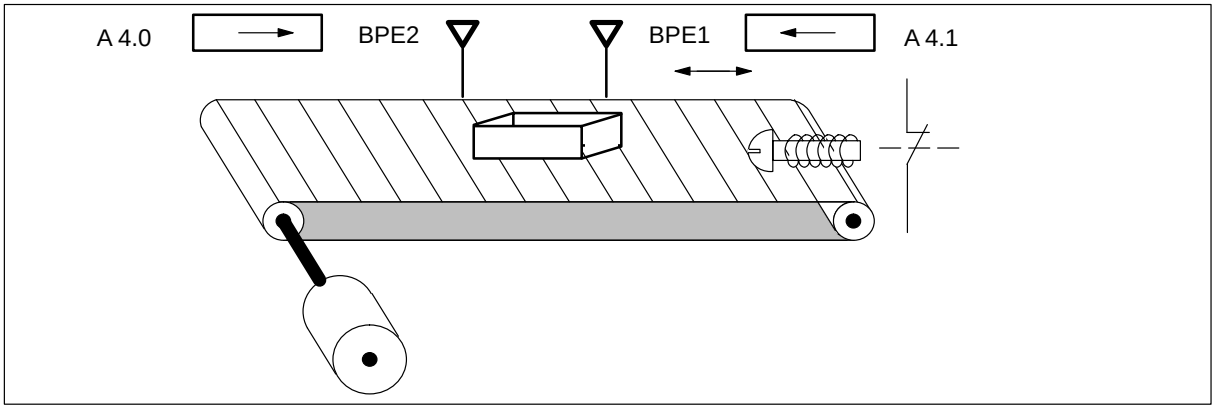
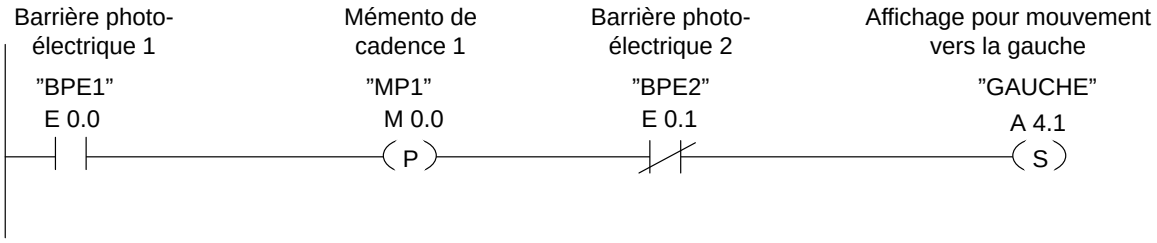


Figure B-3 Système convoyeur avec barrières photoélectriques pour détecter le sens de déplacement

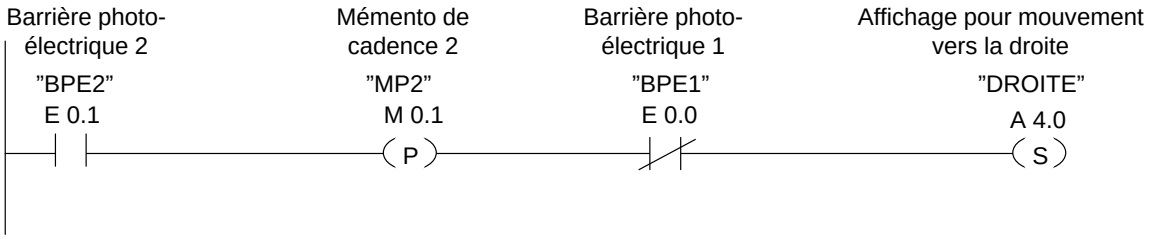
Tableau B-4 Eléments de programmation absolue pour la détection du sens de déplacement

Composante du système	Désignation absolue
Barrière photoélectrique 1	E 0.0
Barrière photoélectrique 2	E 0.1
Affichage pour mouvement vers la droite	A 4.0
Affichage pour mouvement vers la gauche	A 4.1
Mémento de cadence 1	M 0.0
Mémento de cadence 2	M 0.1

Réseau 1 : Si l'état de signal à l'entrée E 0.0 passe de 0 à 1 (front montant) et si l'état de signal à l'entrée E 0.1 est simultanément à 0, le paquet sur le tapis se déplace vers la gauche.



Réseau 2 : Si l'état de signal à l'entrée E 0.1 passe de 0 à 1 (front montant) et si l'état de signal à l'entrée E 0.0 est simultanément à 0, le paquet sur le tapis se déplace vers la droite. Si l'une des barrières photoélectriques est interrompue, cela signifie qu'un paquet se trouve entre les deux barrières.



Réseau 3 : Si une des barrières photoélectriques est interrompue, un paquet se trouve entre les barrières. L'indicateur de sens se désactive.

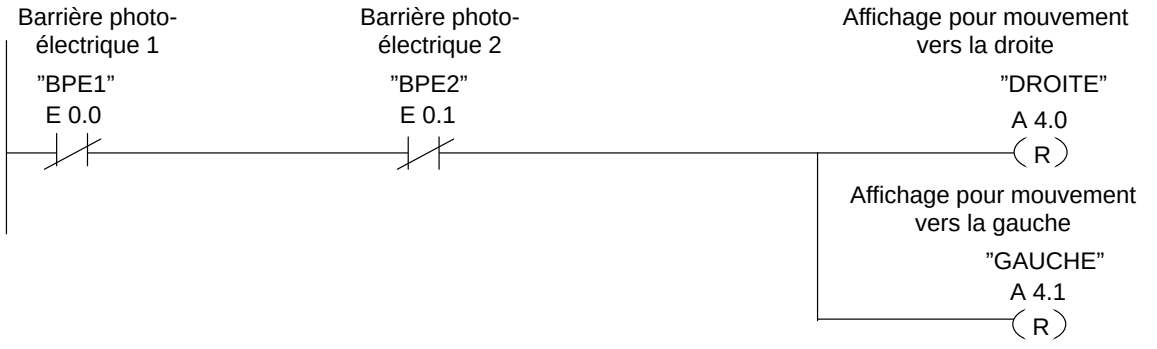


Figure B-4 Schéma à contacts pour détecter le sens de déplacement d'un tapis roulant

B.3 Opérations de temporisation

Générateur d'horloge

Vous pouvez utiliser, pour produire un signal qui se répète périodiquement, un générateur d'impulsions d'horloge ou un relais clignotant. On trouve souvent des générateurs d'horloge dans les systèmes de signalisation qui commandent le clignotement des lampes de signalisation.

Dans l'automate S7-300, vous pouvez réaliser la génération d'impulsions d'horloge en utilisant le traitement commandé par horloge dans des blocs d'organisation spéciaux. Toutefois, l'exemple présenté dans le programme CONT suivant illustre l'utilisation de fonctions de temporisation pour générer une période d'horloge.

L'exemple suivant montre comment réaliser un générateur d'horloge en roue libre à l'aide d'une temporisation (rapport d'impulsion 1:1). La fréquence est subdivisée dans les valeurs présentées au tableau B-5.

Réseau 1 : Si l'état de signal de la temporisation T1 est 0, charger la valeur de temps 250 ms dans T1 et démarrer T1 sous forme d'impulsion prolongée.



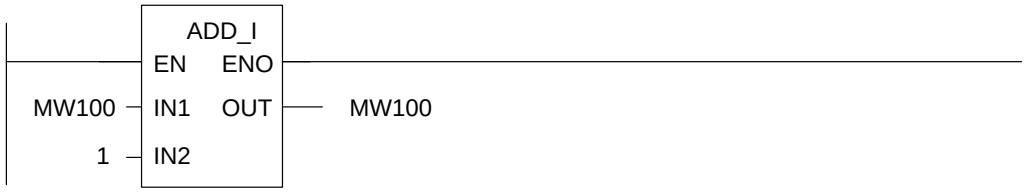
Réseau 2 : L'état de la temporisation est provisoirement mémorisé dans un mémento auxiliaire.



Réseau 3 : Si l'état de signal de la temporisation T est 1, sauter au repère de saut M001.



Réseau 4 : Le mot de mémento MW100 est incrémenté de 1 à chaque fois que la temporisation s'est écoulée.



Réseau 5 : L'opération MOVE vous permet de voir les différentes fréquences d'horloge aux sorties A 12.0 à A 13.7.

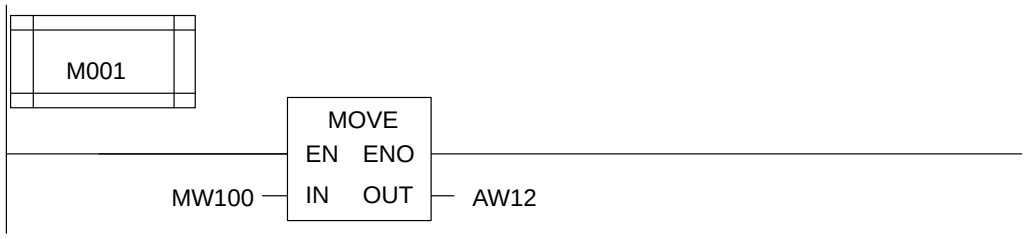


Figure B-5 Schéma à contact pour générer une période d'horloge

L'interrogation de l'état de signal de la temporisation T1 fournit le résultat logique (RLG, voir paragraphe 6.2) présenté à la figure B-6.

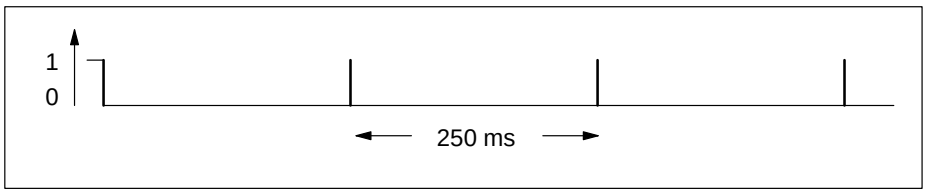


Figure B-6 RLG pour le contact T1 inversé dans l'exemple de période d'horloge

La temporisation est redémarrée une fois le temps écoulé. De ce fait, l'interrogation de l'état de signal par l'opération —|/|— T1 ne délivre l'état de signal 1 que brièvement.

La figure B-7 montre comment se présente le bit RLG inversé.

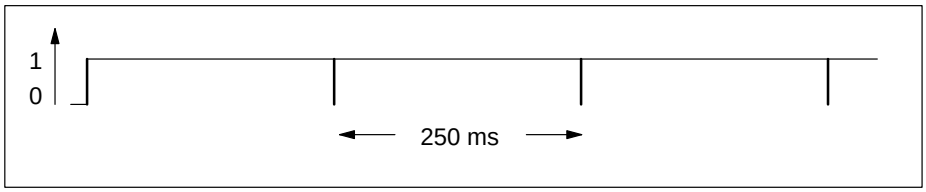


Figure B-7 Bit RLG inversé de la temporisation T1 dans l'exemple de période d'horloge

Le bit RLG est égal à 0 toutes les 250 ms. Le saut est ignoré et le contenu du mot de mémento MW100 est incrémenté de 1.

Obtenir une fréquence précise

Le tableau B-5 énumère les fréquences que vous pouvez obtenir à partir des différents bits de l'octet de mémento MB101 et MB100. Le réseau 4 dans le schéma à contacts représenté à la figure B-5 montre comment l'opération MOVE vous permet de voir les différentes fréquences d'horloge aux sorties A 12.0 à A 13.7.

Tableau B-5 Fréquences pour l'exemple de période d'horloge

Bits de MB101/MB100	Fréquence en hertz	Durée
M 101.0	2,0	0,5 s (250 ms marche/250 ms arrêt)
M 101.1	1,0	1 s (0,5 s marche/0,5 s arrêt)
M 101.2	0,5	2 s (1 s marche/1 s arrêt)
M 101.3	0,25	4 s (2 s marche/2 s arrêt)
M 101.4	0,125	8 s (4 s marche/4 s arrêt)
M 101.5	0,0625	16 s (8 s marche/8 s arrêt)
M 101.6	0,03125	32 s (16 s marche/16 s arrêt)
M 101.7	0,015625	64 s (32 s marche/32 s arrêt)
M 100.0	0,0078125	128 s (64 ms marche/64 ms arrêt)
M 100.1	0,0039062	256 s (128 s marche/128 s arrêt)
M 100.2	0,0019531	512 s (256 s marche/256 s arrêt)
M 100.3	0,0009765	1024 s (512 s marche/512 s arrêt)

Tableau B-5 Fréquences pour l'exemple de période d'horloge (suite)

Bits de MB101/MB100	Fréquence en hertz	Durée
M 100.4	0,0004882	2048 s (1024 s marche/1024 s arrêt)
M 100.5	0,0002441	4096 s (2048 s marche/2048 s arrêt)
M 100.6	0,000122	8192 s (4096 s marche/4096 s arrêt)
M 100.7	0,000061	16384 s (8192 s marche/8192 s arrêt)

Le tableau B-6 énumère les états de signaux des bits de l'octet de mémento MB101. La figure B-8 montre l'état de signal du bit de mémento M101.1.

Tableau B-6 Etat de signal des bits de l'octet de mémento MB101

Cycle	Etat de signal des bits de l'octet de mémento MB101								Valeur de temps [ms]
	7	6	5	4	3	2	1	0	
0	0	0	0	0	0	0	0	0	250
1	0	0	0	0	0	0	0	1	250
2	0	0	0	0	0	0	1	0	250
3	0	0	0	0	0	0	1	1	250
4	0	0	0	0	0	1	0	0	250
5	0	0	0	0	0	1	0	1	250
6	0	0	0	0	0	1	1	0	250
7	0	0	0	0	0	1	1	1	250
8	0	0	0	0	1	0	0	0	250
9	0	0	0	0	1	0	0	1	250
10	0	0	0	0	1	0	1	0	250
11	0	0	0	0	1	0	1	1	250
12	0	0	0	0	1	1	0	0	250

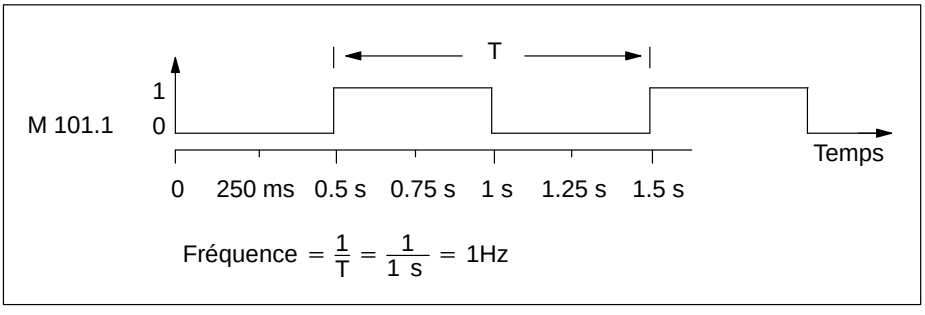


Figure B-8 Etat de signal du bit 1 du MB101 (M 101.1)

B.4 Opérations de comptage et de comparaison

Espace de stockage avec compteur et comparateur

La figure B-9 montre un système avec deux tapis roulants et un espace de stockage temporaire entre eux. La tapis roulant 1 transporte les paquets dans l'espace de stockage. Une barrière photoélectrique à l'extrémité du tapis roulant 1, près de l'espace de stockage, détermine le nombre de paquets qui y sont amenés. Le tapis roulant 2 transporte les paquets de l'espace de stockage temporaire à une rampe de chargement d'où ils sont chargés dans des camions afin d'être livrés aux clients. Une barrière photoélectrique à l'extrémité du tapis roulant 2 près de l'espace de stockage détermine le nombre de paquets transportés de l'espace de stockage à la rampe de chargement.

Un tableau d'affichage avec cinq lampes indique le niveau de remplissage de l'espace de stockage temporaire. La figure B-10 montre le programme CONT pour activer les lampes de signalisation sur le tableau d'affichage.

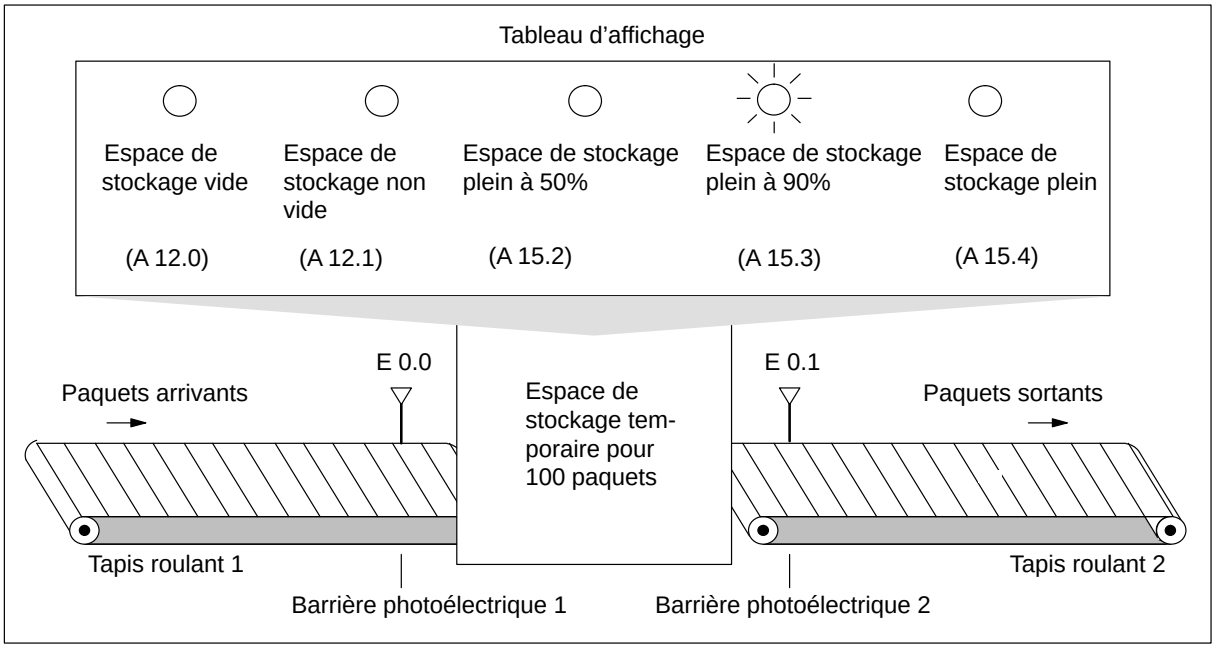
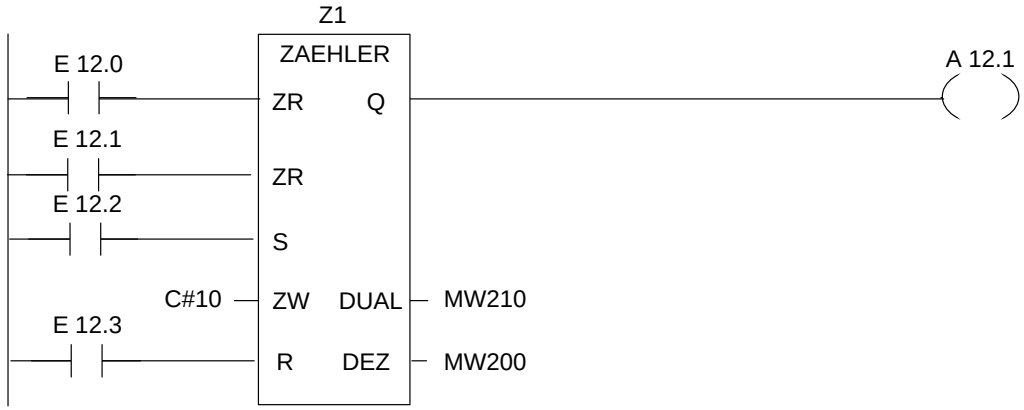
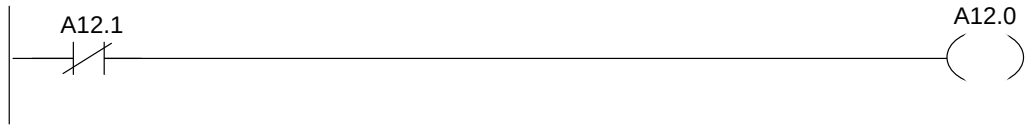


Figure B-9 Espace de stockage avec compteur et comparateur

Réseau 1 : En présence d'un front montant à l'entrée ZV, la valeur du compteur Z1 est augmentée de 1 ; en présence d'un front descendant à l'entrée ZR, elle est diminuée de 1. En présence d'un front montant à l'entrée S, la valeur du compteur est mise à la valeur de ZW. En présence d'un front montant à l'entrée R, la valeur du compteur est remise à zéro. La valeur actuelle du compteur Z1 est mémorisée dans le mot de mémento MW200. La lampe de signalisation A 12.1 indique : "Espace de stockage non vide".



Réseau 2 : La lampe de signalisation A 12.0 indique : "Espace de stockage vide".



Réseau 3 : Si la valeur 50 est inférieure ou égale à la valeur du compteur (c'est-à-dire que la valeur de comptage est supérieure ou égale à 50), la lampe de signalisation "Espace de stockage plein à 50 %" s'allume.



Réseau 4 : Si la valeur du compteur est supérieure ou égale à 90, la lampe de signalisation "Espace de stockage plein à 90 %" s'allume.



Réseau 5 : Si la valeur du compteur est supérieure ou égale à 100, la lampe de signalisation "Espace de stockage plein" s'allume. Utilisez la sortie A 4.4 pour bloquer le tapis roulant 1.



Figure B-10 Schéma à contacts pour activer les lampes de signalisation sur un tableau d'affichage

B.5 Opérations arithmétiques sur nombres entiers

Calcul d'une équation

L'exemple de programme de la figure B-11 vous montre comment utiliser trois opérations arithmétiques sur nombres entiers et les opérations L et T pour obtenir le même résultat qu'avec l'équation suivante :

$$MW4 = \frac{(EW0 + DBW3) \times 15}{MW0}$$

Réseau 1 : Ouvrir bloc de données DB1



Réseau 2 : Le mot d'entrée EW0 est additionné au mot de données global DBW3 (le bloc de données doit avoir été défini et ouvert) et la somme est chargée dans le mot de mémento MW100. MW100 est ensuite multiplié par 15 et le résultat mémorisé dans le mot de mémento MW102. Puis, MW102 est divisé par MW0 et le résultat mémorisé dans MW4. Tant que tous les résultats sont compris dans la plage autorisée pour chaque opération, la sortie de validation ENO transmet un état de signal égal à 1 au pavé suivant.

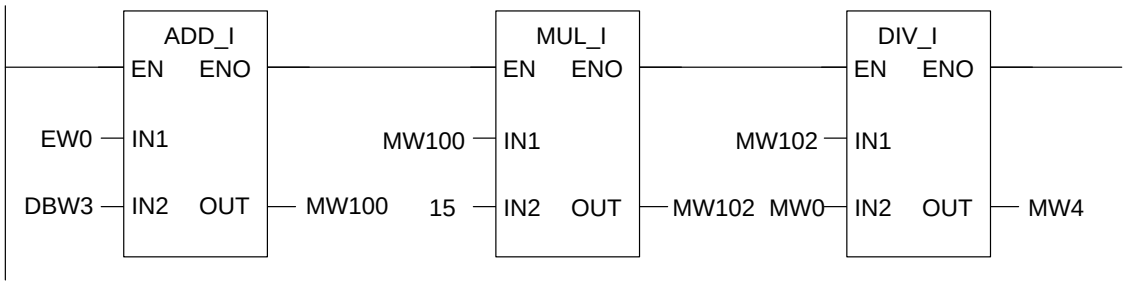


Figure B-11 Schéma à contacts pour opérations arithmétiques sur nombres entiers

B.6 Opérations combinatoires sur mots

Chauffage d'un four

L'opérateur du four montré dans la figure B-12 déclenche le chauffage du four en appuyant sur le bouton-poussoir Marche. Il peut régler la durée du chauffage à l'aide des molettes représentées dans la figure. La valeur indiquée donne les secondes en format décimal codé binaire (DCB). Le tableau B-7 présente les composantes du système de chauffage et les adresses absolues correspondantes, utilisées dans l'exemple de programme représenté à la figure B-12.

Tableau B-7 Composantes du système de chauffage et adresses absolues correspondantes

Composante du système	Adresse absolue dans le programme CONT
Bouton-poussoir Marche	E 0.7
Molette de réglage des unités	E 1.0 à E 1.3
Molette de réglage des dizaines	E 1.4 à E 1.7
Molette de réglage des centaines	E 0.0 à E 0.3
Déclenchement du chauffage	A 4.0

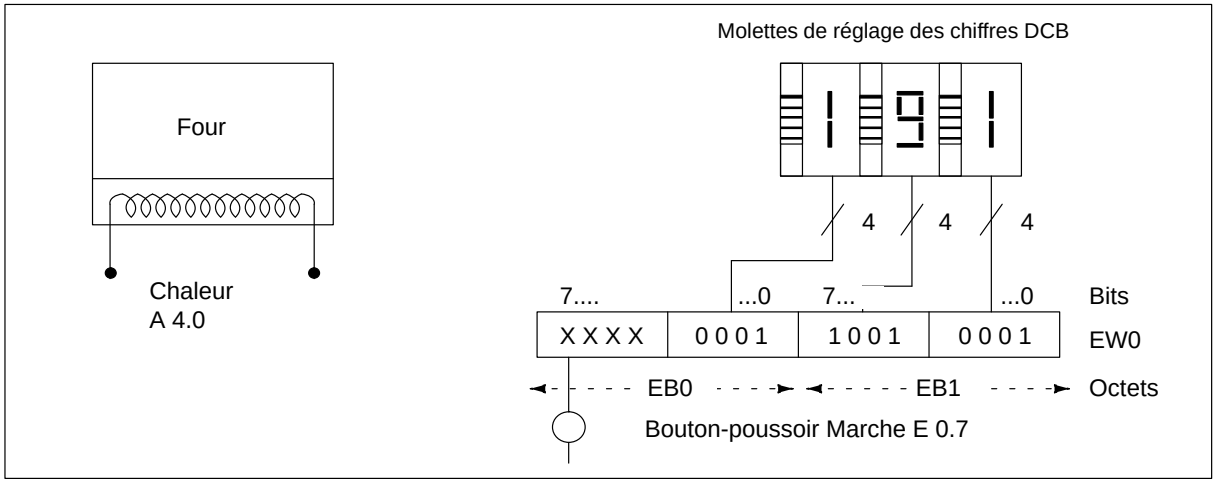


Figure B-12 Utilisation des entrées et sorties pour chauffage à durée limitée

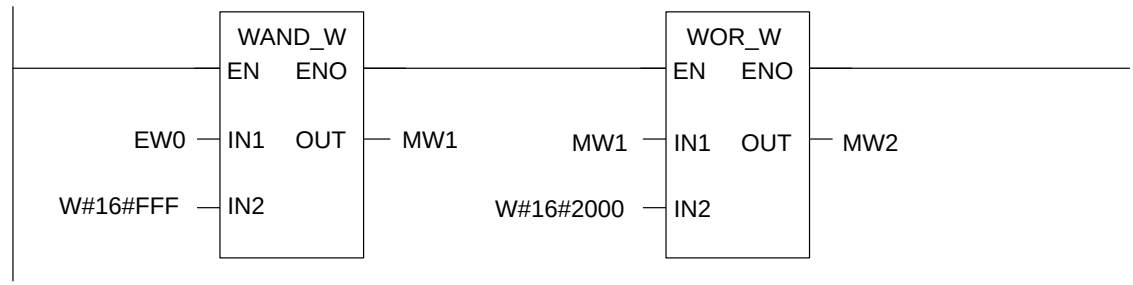
Réseau 1 : Si la temporisation s'exécute, déclencher le chauffage.



Réseau 2 : Si la temporisation s'exécute, l'opération Retour met fin au processus ici.



Réseau 3 : Masquer les bits d'entrée E 0.4 à E 0.7 (c'est-à-dire les mettre à 0). Ces bits d'entrée des molettes ne sont pas utilisés. Les 16 bits des entrées correspondant aux molettes sont combinés à W#16#0FFF avec l'opération ET mot. Le résultat est chargé dans le mot de mémento MW1. Afin de régler la valeur de temps en secondes, la valeur prédéfinie est combinée à W#16#2000 avec l'opération OU mot. Le bit 13 est mis à 1 et le bit 12 est mis à 0.



Réseau 4 : Démarrer la temporisation T1 sous forme d'impulsion prolongée si le bouton-poussoir Marche est enfoncé, en chargeant le mot de mémento MW2 (résultant de la combinaison précédente) comme présélection.



Figure B-13 Schéma à contacts pour le chauffage d'un four

Représentation des nombres

C

Contenu de cette annexe

Paragraphe	Thème	Page
C.1	Représentation des nombres	C-2

C.1 Représentation des nombres

Informations générales

Certaines opérations CONT utilisent des objets de données de taille précise comme illustré au tableau C-2. Les opérations de combinaison sur bits, par exemple, traitent des bits et les opérations de transfert des octets, des mots et des doubles mots.

Les opérations arithmétiques traitent également des octets, des mots et des doubles mots. Dans ces opérandes sous forme d'octets, de mots ou de doubles mots, vous pouvez codifier des nombres de différents formats, comme par exemple les nombres entiers et les nombres réels.

Si vous utilisez l'adressage symbolique, vous définissez des mnémoniques et précisez un type de données pour ces mnémoniques (voir tableau C-2). Les différents types de données présentent des formats et des représentations numériques différents. Les paragraphes suivants contiennent des informations à ce sujet.

Seules quelques-unes des représentations numériques et des représentations de constantes sont décrites dans ce chapitre du manuel.

Tableau C-1 Formats des nombres et constantes qui ne sont pas décrits dans ce chapitre

Format	Taille en bits	Notation
Hexadécimal	8, 16 et 32	B#16#, W#16# et DW#16#
Binaire	8, 16 et 32	2#
Date CEI	16	D#
Durée CEI	32	T#
Heure du jour	32	TOD#
Caractère	8	'A'

Bits, octets, mots et doubles mots

Un bit est un nombre binaire (0 ou 1) ; un octet comprend 8 bits, un mot 16 bits et un double mot 32 bits.

Types de données

Chaque paramètre d'entrée et de sortie d'un pavé CONT correspond à un des types de données suivants :

- type de données élémentaire (voir tableau C-2)
- type de données complexe (tableau (ARRAY), structure (STRUCT), chaîne (STRING), date et heure (DATE_AND_TIME))
- temporisation (TIMER), compteur (COUNTER) et type de bloc
- pointeur (POINTER) et ANY

Pour plus d'informations sur les structures et les tableaux de données que vous pouvez définir, ainsi que sur les autres types de données complexes tels que STRING et DATE_AND_TIME, reportez-vous au manuel de programmation **/120/** et au guide de l'utilisateur **/231/**.

Tableau C-2 Formats constants des types de données élémentaires

Type et description	Taille en bits	Formats	Plage et représentation des nombres (valeur inférieure à valeur supérieure)	Exemple
BOOL (bit)	1	Texte booléen	TRUE/FALSE	TRUE
BYTE (octet)	8	Nombre hexadécimal	B#16#0 à B#16#FF	B#16#10 byte#16#10
WORD (mot)	16	Nombre binaire Nombre hexadécimal Nombre DCB Nombre décimal non signé	2#0 à 2#1111_1111_1111_1111 W#16#0 à W#16#FFFF C#0 à C#999 B#(0,0) à B#(255,255)	2#0001_0000_0000_0000 W#16#1000 word16#1000 C#998 B#(10,20) byte#(10,20)
DWORD (double mot)	32	Nombre binaire Nombre hexadécimal Nombre décimal non signé	2#0 à 2#1111_1111_1111_1111_ 1111_1111_1111_1111 DW#16#0000_0000 à DW#d16#FFFF_FFFF B#(0,0,0,0) à B#(255,255,255,255)	2#1000_0001_0001_1000_ 1011_1011_0111_1111 DW#16#00A2_1234 dword#16#00A2_1234 B#(1, 14, 100, 120) byte#(1,14,100,120)
INT (entier de 16 bits)	16	Nombre décimal signé	-32768 à 32767	1
DINT (entier de 32 bits ou double)	32	Nombre décimal signé	L#–2147483648 à L#2147483647	L#1
REAL (réel)	32	Nombre à virgule flottante IEEE	Limite supérieure : ±3.402823e+38 Limite inférieure : ±1.175 495e-38 (voir aussi tableau C-5)	1,234567e+13
S5TIME (durée SIMATIC)	16	Durée S5 en unités de 10 ms (présélection)	S5T#0H_0M_0S_10MS à S5T#2H_46M_30S_0MS et S5T#0H_0M_0S_0MS	S5T#0H_1M_0S_0MS S5Time#0H_1H_1M_0S_0MS
TIME (durée CEI)	32	Durée CEI en unités de 1 ms, nombre entier signé	T#-24D_20H_31M_23S_648MS à T#24D_20H_31M_23S_647MS	T#0D_1H_1M_0S_0MS TIME#0D_1H_1M_0S_0MS
DATE (date CEI)	16	Date CEI en unités de 1 jour	D#1990-1-1 à D#2168-12-31	D#1994-3-15 DATE#1994-3-15
TIME_OF_ DAY (heure du jour)	32	Heure du jour en unités de 1 ms	TOD#0:0:0.0 à TOD #23:59:59.999	TOD#1:10:3.3 TIME_OF_DAY#1:10:3.3
CHAR (caractère)	8	Caractère ASCII	’A’,’B’ etc..	’E’

Nombres entiers de 16 bits

Un nombre entier a un signe indiquant s’il est positif ou négatif. Un nombre entier de 16 bits occupe un mot en mémoire. Le tableau C-3 montre la plage d’un nombre entier de 16 bits et la figure C-1 présente l’entier +44 en format binaire.

Tableau C-3 Plage des nombres entiers de 16 bits

Format	Plage
Entier de 16 bits	-32 768 à +32 767

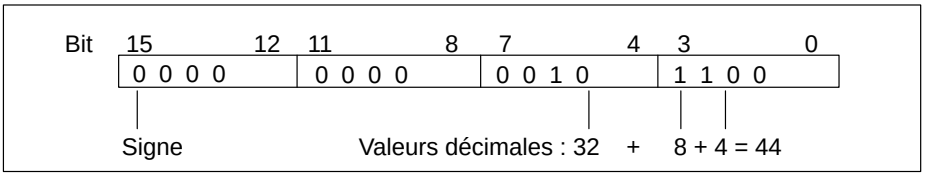


Figure C-1 Entier de 16 bits en format binaire : +44

Nombres entiers de 32 bits

Un nombre entier a un signe indiquant s'il est positif ou négatif. Un nombre entier de 32 bits (entier double) occupe deux mots en mémoire. Le tableau C-4 montre la plage d'un tel entier et la figure C-2 présente l'entier –500 000 en format binaire. Dans ce format, un nombre entier négatif est représenté par le complément à deux de l'entier positif correspondant. On forme ce complément à deux en inversant l'état de signal de tous les bits, puis en ajoutant 1 au résultat.

Tableau C-4 Plage des nombres entiers de 32 bits

Format	Plage
Entier de 32 bits	-2 147 483 648 à +2 147 483 647

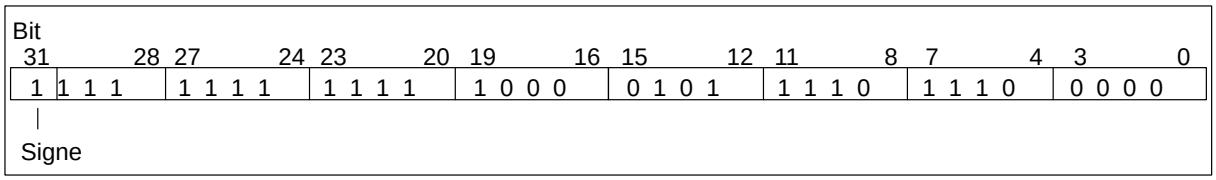


Figure C-2 Entier de 32 bits en format binaire : –500 000

Nombres réels

Un nombre réel – également nommé nombre à virgule flottante – est un nombre positif ou négatif comportant une valeur après la virgule, par exemple 0,339 ou -11,1. Vous pouvez également préciser un exposant pour un nombre réel. Cet exposant correspond à la puissance entière de 10 par laquelle multiplier le nombre réel afin d'obtenir la valeur que vous désirez représenter. Vous pouvez, par exemple, représenter 1 234 000 par 1,234E6 ou 1,234e6 (c'est-à-dire 1,234 * 10^6). Le tableau C-5 montre la plage d'un nombre réel.

Un nombre à virgule flottante occupe deux mots en mémoire (32 bits ; voir figure C-3). Le bit de poids fort – le bit 31 – correspond au signe : 0 pour un nombre réel positif et 1 pour un nombre réel négatif. Les autres bits représentent l'exposant et la mantisse.

Tableau C-5 Plage des nombres réels

Format	Plage [1]
Nombres réels	-3,402 823E+38 à -1,175 495E-38 et ± 0 et +1,175 495E-38 à +3,402 823E+38

1 Si le résultat d'une opération à virgule flottante se situe dans les plages –1,175 495E-38 à –1,401 298E-45 ou +1,401 298E-45 à +1,175 495E-38, un dépassement bas est généré. Il s'agit de la plage des nombres dénormalisés (voir tableau 12-6).

Format des nombres réels

Le format des nombres réels en CONT est le format de base, de simple largeur, décrit dans la norme ANSI/IEEE Std 754–1985, *IEEE Standard for Binary Floating-Point Arithmetic*. Dans ce format, vous ne pouvez représenter que les valeurs définies par les trois paramètres entiers suivants :

- p = nombre de bits significatifs (précision)
- E_{max} = exposant maximum
- E_{min} = exposant minimum

Le tableau C-6 indique les paramètres du format.

Tableau C-6 Paramètres du format des nombres réels

Désignation du paramètre	Valeur du paramètre
p	24
E_{max}	+127
E_{min}	-126
Bias exposant	+127
Largeur d'exposant en bits	8
Largeur du format en bits	32

Le format comprend les entités suivantes :

- Nombres de la forme $(-1)^s\ 2^E\ (b_0 . b_1\ b_2 ... b_{p-1})$, où
  - s = 0 ou 1
  - E = nombre entier de 16 bits quelconque compris entre E_{min} et E_{max} inclus
  - b_i = 0 ou 1
- Deux nombres infinis, + ∞ et - ∞
- Au minimum un NaN signalant (NaN = « not a floating-point number » qui signifie « pas un nombre réel »)
- Au minimum un NaN non signalant

Champs des composants d'un nombre réel

Les nombres réels en format de base, de simple largeur, sont composés des champs suivants (voir figure C-3) :

- un signe s de 1 bit
- un exposant avec bias : e = E + bias
- une fraction : f = **.** $b_1\, b_2 ... b_{p-1}$

La plage de l'exposant E non polarisé englobe tous les nombres entiers de 16 bits compris entre E_{min} et E_{max} (c'est-à-dire –126 à +127), ainsi que deux autres valeurs réservées E_{min} –1 pour le codage de ±0 et des nombres dénormalisés et E_{max} +1 pour le codage de $\pm\infty$ et des NaN.

La figure C-3 montre les trois champs (s, e et f) d'un nombre réel de 32 bits. Dans la figure, un nombre réel X (32 bits) a la valeur v obtenue de la manière suivante à partir des champs qui la composent :

- Si e = 255 et si f $\neq$ 0, alors v est égal à NaN, quel que soit le signe s.
- Si e = 255 et si f = 0, alors v = $(-1)^s \ \infty$.
- Si 0 < e < 255, alors v = $(-1)^s\, 2^{e-127}\, (1\ .\ f)$

  (Il s'agit d'un nombre dénormalisé)
- Si e = 0 et f $\neq$ 0, alors v = $(-1)^s\, 2^{e-126}\, (0\ .\ f)$.

  (Il s'agit d'un nombre dénormalisé)
- Si e = 0 et f = 0, alors v = $(-1)^s$ 0 (zéro).

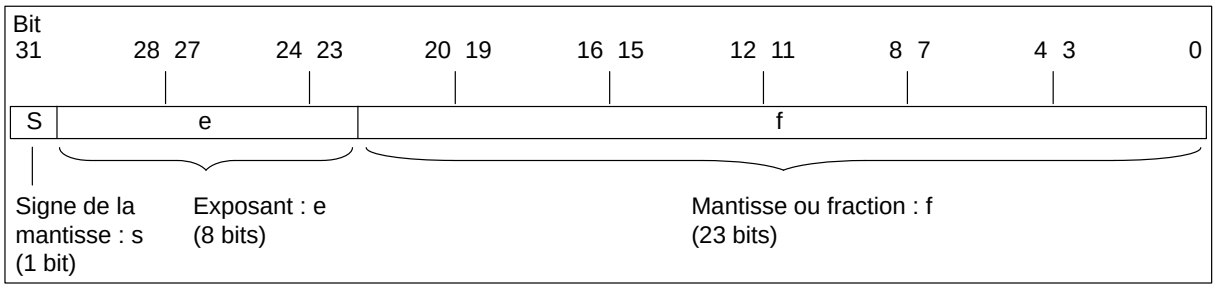


Figure C-3 Format d'un nombre réel

Exemple du format des nombres réels

La figure C-4 montre le format des nombres réels correspondant aux valeurs décimales suivantes :

- 10,0
- π (3,141593)
- racine carrée de 2 ($\sqrt{2}$ = 1,414214)

La valeur hexadécimale des nombres réels figure au-dessus du numéro de bit.

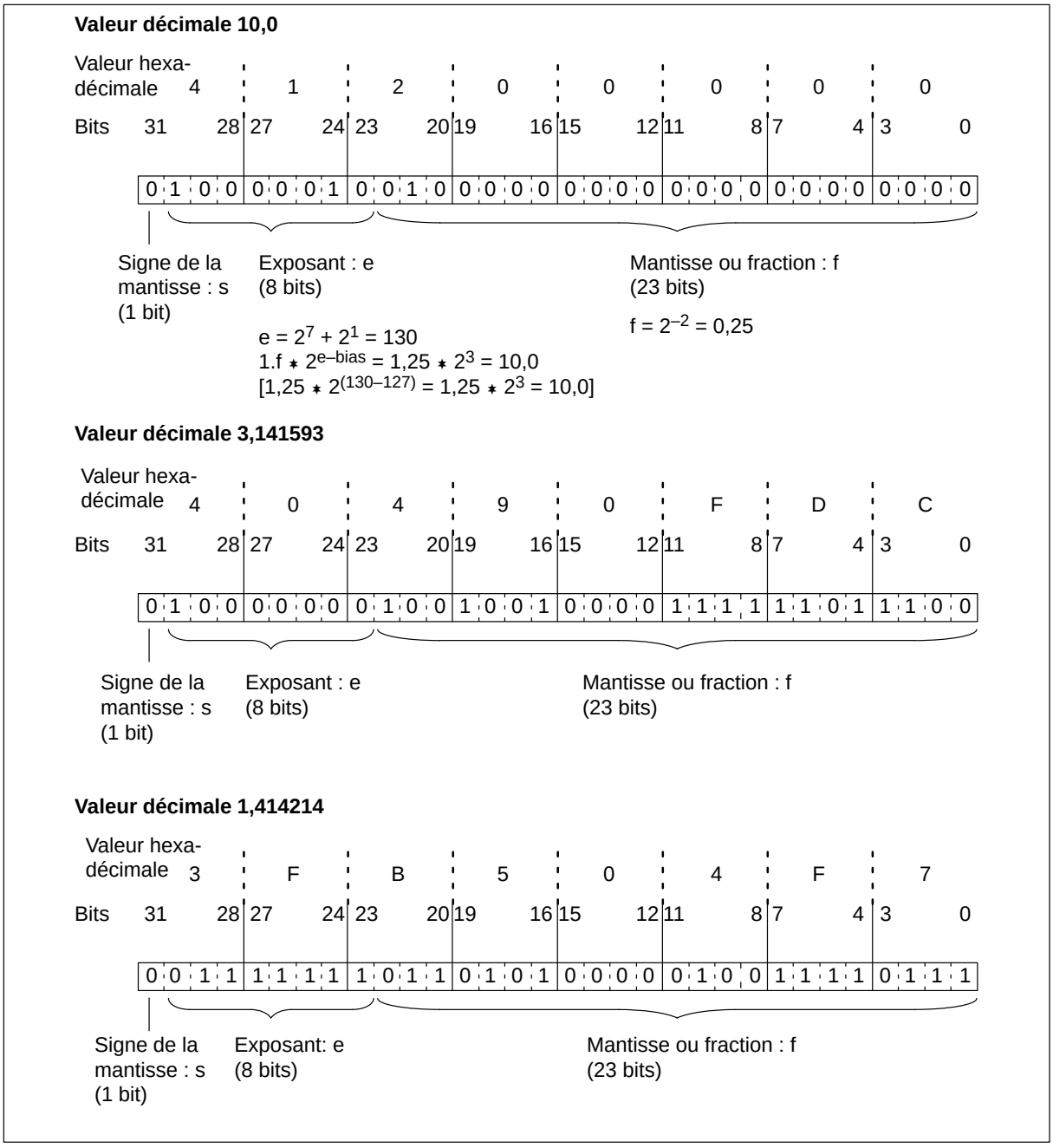

Valeur décimale 10,0
Valeur hexadécimale 4 1 2 0 0 0 0 0
Bits 31 28 27 24 23 20 19 16 15 12 11 8 7 4 3 0
0 1 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Signe de la mantisse : s (1 bit)
Exposant : e (8 bits)
e = 2^7 + 2^1 = 130
1.f * 2e–bias = 1,25 * 2^3 = 10,0
[1,25 * 2^(130–127) = 1,25 * 2^3 = 10,0]
Mantisse ou fraction : f (23 bits)
f = 2^–2 = 0,25
Valeur décimale 3,141593
Valeur hexadécimale 4 0 4 9 0 F D C
Bits 31 28 27 24 23 20 19 16 15 12 11 8 7 4 3 0
0 1 0 0 0 0 0 0 0 1 0 0 1 0 0 1 0 0 0 0 1 1 1 1 1 1 0 1 1 1 0 0
Signe de la mantisse : s (1 bit)
Exposant : e (8 bits)
Mantisse ou fraction : f (23 bits)
Valeur décimale 1,414214
Valeur hexadécimale 3 F B 5 0 4 F 7
Bits 31 28 27 24 23 20 19 16 15 12 11 8 7 4 3 0
0 0 1 1 1 1 1 1 1 0 1 1 0 1 0 1 0 0 0 0 0 1 0 0 1 1 1 1 0 1 1 1
Signe de la mantisse : s (1 bit)
Exposant: e (8 bits)
Mantisse ou fraction : f (23 bits)

Figure C-4 Exemple du format de nombres réels pour la valeur décimale 10,0

Nombres décimaux codés binaires

Dans le format décimal codé binaire (DCB), un nombre décimal est représenté à l'aide de groupes de bits. Un groupe de 4 bits correspond à un chiffre du nombre décimal signé ou au signe de ce nombre. Les groupes de 4 bits forment un mot (16 bits) ou un double mot (32 bits). Les quatre bits de poids fort précisent le signe du nombre : 0000 pour un nombre positif et 1111 pour un nombre négatif. Les commandes avec opérandes décimaux codés binaires analysent uniquement le bit de poids fort (bit 15 dans le cas d'un mot, bit 31 dans le cas d'un double mot). Le tableau C-7 montre le format et la plage des deux types de nombres DCB. Les figures C-5 et C-6 donnent un exemple de nombre décimal codé binaire, respectivement en format de mot et en format de double mot.

Tableau C-7 Nombres décimaux codés binaires de 16 et de 32 bits

Format	Plage
Mot (16 bits, nombre DCB à trois chiffres avec signe)	–999 à +999
Double mot (32 bits, nombre DCB à sept chiffres avec signe)	–9 999 999 à +9 999 999

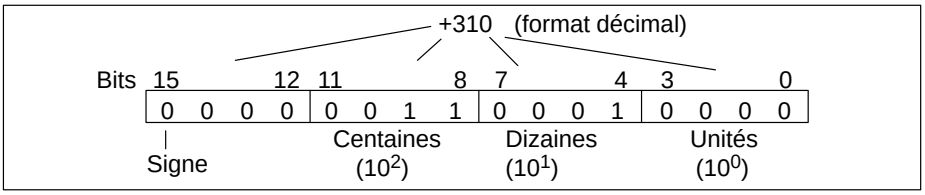


Figure C-5 Nombre décimal codé binaire en format de mot

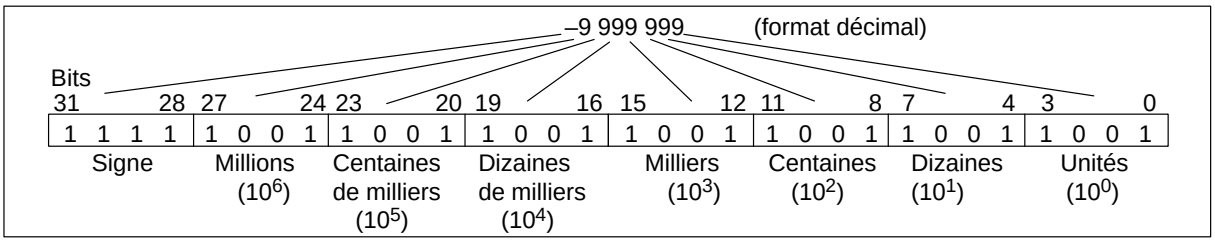

–9 999 999 (format décimal)
Bits
31 28 27 24 23 20 19 16 15 12 11 8 7 4 3 0
1 1 1 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1
Signe
Millions (10^6)
Centaines de milliers (10^5)
Dizaines de milliers (10^4)
Milliers (10^3)
Centaines (10^2)
Dizaines (10^1)
Unités (10^0)

Figure C-6 Nombre décimal codé binaire en format de double mot

Saisie de la durée

Lorsque vous indiquez une durée avec le type de données S5TIME, vos entrées sont mémorisées en format décimal codé binaire (DCB ; voir figure C-7 et tableau C-8).

Avec S5TIME, vous précisez une valeur de temps de 0 à 999 ainsi qu'une base de temps (voir tableau C-8). La base de temps correspond à l'intervalle par lequel une temporisation décrémente la valeur de temps d'une unité jusqu'à ce que cette valeur de temps atteigne 0.

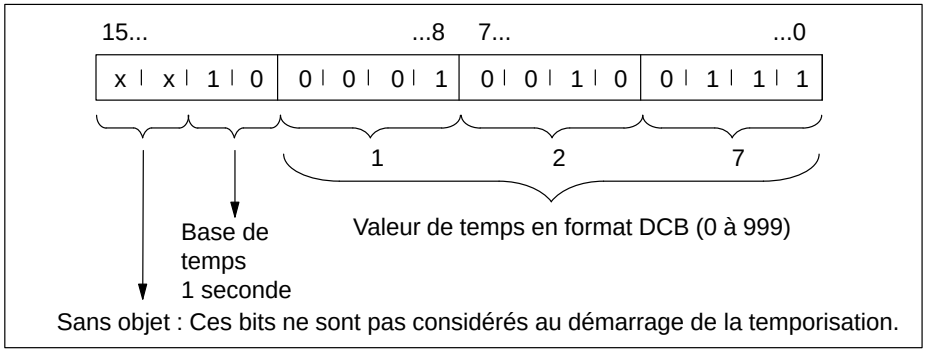


Figure C-7 Contenu de l'opérande de temporisation : valeur de temps 127, base de temps 1 s

Tableau C-8 Base de temps pour S5TIME

Base de temps	Code binaire pour la base de temps
10 ms	00
100 ms	01
1 s	10
10 s	11

Vous pouvez charger une valeur de temps prédéfinie en utilisant la syntaxe suivante :

- W#16#wxyz
  - avec : w = base de temps (c'est-à-dire intervalle de temps ou résolution)
  - xyz = valeur de temps en format DCB
- S5T#aH_bbM_ccS_dddMS
  - avec : a = heures, bb = minutes, cc = secondes et ddd = millisecondes
  - La base de temps est sélectionnée automatiquement et la valeur est arrondie au nombre inférieur le plus proche avec cette base de temps.

Vous pouvez indiquer une valeur de temps de 9 990 secondes ou de 2H_46M_30S au maximum.

Saisie de la date et de l'heure

Lorsque vous indiquez une date et une heure avec le type de données DATE_AND_TIME, vos entrées sont mémorisées en format décimal codé binaire (voir tableau C-9). Le type de données DATE_AND_TIME possède la plage suivante :

DT#1990-1-1-0:0:0.0 à DT#2089-12-31-23:59:59.999

Les exemples suivants vous montrent la syntaxe permettant de saisir la date et l'heure pour le jeudi 25 décembre 1993, à 8 heures 1 minute et 1,23 seconde. Les deux formats suivants sont possibles :

- DATE_AND_TIME#1993-12-25-8:01:1.23
- DT#1993-12-25-8:01:1.23

Les fonctions standard CEI (Commission Electrotechnique Internationale) suivantes sont disponibles pour être utilisées avec le type de données DATE_AND_TIME (pour de plus amples informations, reportez-vous au manuel de programmation **/234/**).

- Conversion de la date et l'heure du jour en format DATE_AND_TIME (DT)

  FC3: D_TOD_DT

- Extraction de la date du format DATE_AND_TIME

  FC6: DT_DATE

- Extraction du jour de la semaine du format DATE_AND_TIME

  FC7: DT_DAY

- Extraction de l'heure du jour du format DATE_AND_TIME

  FC8: DT_TOD

Le tableau C-9 présente le contenu des octets où se trouvent les informations d'horodatage pour le jeudi 25 décembre 1993, à 8 heures 1 minute 1,23 seconde.

Tableau C-9 Contenu des octets d'horodatage

Octet	Contenu	Exemple
0	Année	B#16#93
1	Mois	B#16#12
2	Jour	B#16#25
3	Heures	B#16#08
4	Minutes	B#16#01
5	Secondes	B#16#01
6	Deux chiffres les plus significatifs de MSEC	B#16#23
7 (4 bits de poids fort)	Chiffre le moins significatif de MSEC	B#16#0
7 (4 bits de poids faible)	Jour de la semaine 1 = dimanche 2 = lundi ... 7 = samedi	B#16#5

Bibliographie

D

/30/ Petit manuel illustré : *Faites connaissance avec le S7-300...*

/70/ Manuel : *Automate programmable S7-300,*
Installation et configuration - Caractéristiques des CPU

/71/ Manuel de référence : *Systèmes d'automatisation S7-300, M7-300*
Caractéristiques des modules

/72/ Liste des opérations : *Automate programmable S7-300*

/100/ Manuel de mise en œuvre : *Systèmes d'automatisation S7-400, M7-400,*
Installation et configuration

/101/ Manuel de référence : *Systèmes d'automatisation S7-400, M7-400*
Caractéristiques des modules

/102/ Liste des opérations : *Automate programmable S7-400*

/231/ Guide de l'utilisateur : *Logiciel de base pour SIMATIC S7 et M7,*
STEP 7

/232/ Manuel : *LIST pour SIMATIC S7-300/400,*
Programmation de blocs

/234/ Manuel de programmation : *Logiciel système pour SIMATIC S7-300/400*
Conception de programmes

/235/ Manuel de référence : *Logiciel système pour SIMATIC S7-300/400*
Fonctions standard et fonctions système

/236/ Manuel : *LOG pour SIMATIC S7-300/400*
Programmation de blocs

/237/ *Index général*, STEP 7

/250/ Manuel : *SCL pour SIMATIC S7-300/400,*
Programmation de blocs

/251/ Manuel : *GRAPH pour SIMATIC S7-300/400,*
Programmation de commandes séquentielles

/252/ Manuel : *HiGraph pour SIMATIC S7-300/400,*
Programmation de graphes d'état

/253/ Manuel : *C pour SIMATIC S7-300/400,*
Programmation en C

/254/ Manuel : *CFC pour SIMATIC S7 et M7,*
Comment câbler graphiquement des fonctions technologiques ?

/270/ Manuel : *Diagnostic processus en CONT, LOG et LIST.*

/271/ Manuel : *NETPRO*
Configuration graphique de réseaux

/800/ *DOCPRO,*
Documentation normalisée d'un projet (uniquement sur CD-ROM)

/801/ *Téléservice pour automates programmables de type S7,C7 et M7*,
(uniquement sur CD-ROM)

/802/ *Simulateur PLC de STEP 7*,
(uniquement sur CD-ROM)

/803/ Manuel de référence : *Logiciel standard pour S7-300/400*,
STEP 7 Fonctions standard. Partie 2 (uniquement sur CD-ROM)

Glossaire

A

Abréviations

Il s'agit des abréviations des opérandes et des opérations de programmation dans un programme (E pour entrée, par exemple). STEP 7 prend en charge les abréviations internationales (basées sur l'anglais) et les abréviations SIMATIC (basées sur l'allemand et sur les conventions en vigueur pour l'adressage SIMATIC).

Accumulateur

Un accumulateur est un registre de la CPU qui sert de mémoire intermédiaire aussi bien pour les opérations de chargement et de transfert que les opérations de comparaison, de calcul et de conversion.

Adressage absolu

Type d'adressage dans lequel l'adresse indiquée est celle de l'opérande à traiter.

Adressage direct

Type d'adressage dans lequel l'opérande d'une opération se réfère directement à l'adresse en mémoire de la valeur sur laquelle l'opération doit porter.

Exemple :
l'adresse A 4.0 indique le bit 0 de l'octet 4 de la mémoire image des sorties.

Adressage immédiat

Type d'adressage dans lequel l'opérande contient la valeur sur laquelle l'opération doit porter.

Exemple : L27 signifie le chargement de la constante 27 dans l'accumulateur.

Adressage symbolique

Type d'adressage dans lequel l'opérande à traiter est indiqué de façon symbolique sous forme de mnémoniques (à la place d'une adresse).

B

BI1 et BI0

Les bits d'état BI1 et BI0 (bits indicateurs) fournissent des informations sur :

- le résultat d'une opération arithmétique,
- le résultat d'une opération de comparaison,
- le résultat d'une opération sur mots ou
- les bits décalés par une opération de décalage ou de rotation.

Bit d'état

Le bit d'état (ETAT) contient la valeur d'un bit en accès. L'état d'une opération de combinaison ayant accès en lecture à la mémoire (U, UN, O, ON, X ou XN) est toujours le même que la valeur du bit interrogé par cette opération (bit sur lequel elle effectue sa combinaison). L'état d'une opération de combinaison ayant accès en écriture à la mémoire (S, R, =) est identique à la valeur du bit dans lequel l'opération écrit ou bien, si aucune écriture n'a lieu, à la valeur du bit auquel accède l'opération. Le bit d'état n'a pas de signification pour les opérations de combinaison qui n'accèdent pas à la mémoire ; ces opérations mettent le bit d'état à 1. Le bit d'état n'est pas interrogé par des opérations, mais uniquement exploité pendant le test du programme (état du programme).

Bit DEB

Le bit d'état DEB signifie « Débordement ». Un débordement peut se produire à la suite de l'exécution d'une fonction arithmétique, par exemple.

Bit DM

Le bit d'état DM signifie « Débordement mémorisé ». Un débordement peut se produire à la suite de l'exécution d'une fonction arithmétique, par exemple.

Bit RB

Le bit de résultat binaire (RB) constitue un lien entre le traitement de bits et le traitement de mots. Il rend possible de manière efficace l'interprétation binaire du résultat d'une opération sur mot et l'intégration de ce résultat à une séquence combinatoire sur bits.

Bloc de code

Pour SIMATIC S7, un bloc de code est un bloc qui comprend une partie du programme utilisateur STEP 7. Contrairement au bloc de code, un bloc de données ne comprend que des données. Parmi les blocs de codes figurent les blocs d'organisation (OB), les blocs fonctionnels (FB), les fonctions (FC), les blocs fonctionnels système (SFB) et les fonctions système (SFC).

Bloc de données (DB)

Un bloc de données (DB) est une zone de données dans un programme utilisateur qui contient des données utilisateur. Il existe des blocs de données globaux accessibles par tous les blocs de code, et des blocs de données d'instance associés à un appel particulier de blocs fonctionnels. Contrairement à tous les autres blocs, les blocs de données ne contiennent pas d'instructions.

Bloc de données d'instance

Un bloc de données d'instance sauvegarde les paramètres formels et les données locales de blocs fonctionnels. Un bloc de données d'instance peut être associé à un appel de FB ou à une hiérarchie d'appel de blocs fonctionnels.

Bloc de données global

Sur le plan technique, un bloc de données global est un bloc de données dont l'opérande est chargé dans le registre des adresses du DB au moment de l'ouverture du bloc. Il fournit de la mémoire et des données pour tous les blocs de code (FC, FB ou OB) à exécuter.

Contrairement au bloc de données global, un bloc de données d'instance est utilisé comme mémoire spéciale et comme données pour le bloc fonctionnel auquel il est associé.

Bloc fonctionnel (FB)

Conformément à la norme CEI 1131-3, un bloc fonctionnel est un bloc de code avec données statiques. Un bloc fonctionnel permet la transmission de paramètres dans le programme utilisateur. Pour cette raison, les blocs fonctionnels sont appropriés pour la programmation de fonctions complexes à caractère répétitif, telles que les régulations, la sélection de mode de fonctionnement.

Bloc fonctionnel système (SFB)

Un bloc fonctionnel système (SFB) est un bloc fonctionnel (avec mémoire) intégré au système d'exploitation de la CPU qui peut être appelé depuis le programme utilisateur STEP 7, si besoin est.

C

Classeur

Dossier de l'interface utiliseur du gestionnaire de projets SIMATIC (SIMATIC Manager) pouvant être ouvert et pouvant contenir d'autres classeurs et objets.

D

Diagramme

Source graphique spéciale, créée à l'aide du langage de programmation CFC.

Données de référence

Les données de référence permettent le contrôle du programme CPU. Elles comprennent la liste des références croisées, le tableau d'affectation, la structure du programme utilisateur, la liste des opérandes non utilisés et celle des opérandes sans mnémonique.

Données statiques

Les données statiques sont des données locales d'un bloc fonctionnel sauvegardées dans un bloc de données d'instance et, donc, conservées jusqu'au traitement suivant du bloc fonctionnel.

F

Fonction (FC)

Conformément à la norme CEI 1131-3, une fonction est un bloc de code sans données statiques. Une fonction permet la transmission de paramètres dans le programme utilisateur. Pour cette raison, les fonctions sont appropriées pour la programmation de fonctions complexes à caractère répétitif, telles que les calculs.

Fonction système (SFC)

Une fonction système (SFC) est une fonction (sans mémoire) intégrée au système d'exploitation de la CPU qui peut être appelée depuis le programme utilisateur STEP 7, si besoin est.

H

Hiérarchie d'appel

Il est nécessaire d'appeler tous les blocs avant de pouvoir les traiter. On appelle hiérarchie d'appel l'ordre et le degré d'imbrication de ces appels au sein d'un bloc d'organisation.

I

Identificateur d'opérande

Un identificateur d'opérande est une partie de l'opérande d'une opération contenant des informations, telles que la zone de mémoire dans laquelle se trouve la valeur (objet de données) ou la taille d'une valeur (objet de données) nécessaire à l'exécution d'une opération de combinaison. Dans l'instruction « U EB10 », l'identificateur d'opérande est « EB », E désignant la zone des entrées et B un octet dans cette zone.

Instance

Par instance, on désigne l'appel d'un bloc fonctionnel. Si un bloc fonctionnel est appelé à 5 reprises dans le programme utilisateur STEP 7, on parle alors de 5 instances. A chaque appel est associé un bloc de données d'instance.

L

Liste d'instructions (LIST)

La liste d'instructions (LIST) est un langage de programmation littéral proche du langage machine. LIST est le langage assembleur de STEP 5 et STEP 7. Dans un programme écrit en LIST, les différentes instructions correspondent aux étapes de traitement du programme par la CPU.

Logigramme (LOG)

Le logigramme est un langage de programmation pour STEP 5 et STEP 7. En LOG, la logique est représentée à l'aide des boîtes logiques de l'algèbre booléenne.

M

Master Control Relay

Voir Relais de masquage

Mnémonique

Un mnémonique est un nom symbolique défini par l'utilisateur obéissant à des règles de syntaxe déterminées. Après avoir indiqué à quoi doit correspondre ce nom (variable, type de données, repère de saut, bloc, par exemple), vous pouvez l'utiliser dans la programmation, ainsi que pour le contrôle-commande. Exemple : opérande : E 5.0, type de données : BOOL, mnémonique : Arrêt_Urg.

Mot d'état

Le mot d'état est un élément constituant du registre de l'unité centrale. Dans le mot d'état sont mémorisées des informations d'état et des informations d'erreur pouvant apparaître dans le cadre du traitement d'instructions STEP 7. Les bits d'état peuvent être lus et traités par l'utilisateur ; les bits d'erreur peuvent uniquement être lus.

Mot-clé

Les mots-clés sont utilisés en mode de saisie source. Lors de la saisie du programme, ils permettent d'identifier le début d'un bloc et de sélectionner les parties de la section de déclaration de blocs, le début des commentaires de blocs et le début des titres.

O

Opérande

Un opérande est la partie d'une instruction STEP 7 indiquant ce avec quoi le processeur doit exécuter l'opération. L'adressage d'un opérande peut être aussi bien absolu que symbolique.

Opération

Une opération est la partie d'une instruction STEP 7 indiquant ce que le processeur doit faire.

P

Paramètre effectif

Les paramètres effectifs remplacent les paramètres formels lors de l'appel d'un bloc fonctionnel (FB) ou d'une fonction (FC).

Exemple : le paramètre effectif « E 3.6 » remplace le paramètre formel « Demarr ».

Paramètre formel

Un paramètre formel réserve la place d'un paramètre effectif dans les blocs de code paramétrables. Dans le cas de blocs fonctionnels et de fonctions, c'est l'utilisateur qui déclare les paramètres formels, alors qu'il le sont déjà pour les blocs fonctionnels système et les fonctions système. Lors de l'appel du bloc, un paramètre effectif est affecté au paramètre formel de sorte que le bloc appelé utilise cette valeur en cours.

Les paramètres formels comptent parmi les données locales du bloc. On distingue les paramètres d'entrée, les paramètres de sortie et les paramètres d'entrée/sortie.

Pointeur

Un pointeur permet d'identifier l'adresse d'une variable. Un pointeur contient une opérande à la place d'une valeur. Lorsque vous affectez un paramètre effectif au type de paramètre POINTER, vous fournissez l'adresse en mémoire. STEP 7 vous permet de saisir le pointeur soit en format de pointeur, soit tout simplement sous forme d'opérande (M 50.0 par exemple). L'exemple suivant illustre le format de pointeur permettant d'accéder à des données à partir de M 50.0 :

P#M50.0

Première interrogation

Première mise à 1 du résultat logique (RLG)

Programme S7

Classeur pour blocs, sources et diagrammes nécessaires aux modules S7 programmables et qui contient également la table des mnémoniques.

Programme utilisateur (objet logiciel)

Classeur pour blocs chargés dans un module S7 programmable (CPU ou FM, par exemple) et exécutables à cet endroit pour commander une installation ou un processus.

Programme utilisateur (terme général)

Un programme utilisateur comprend toutes les instructions et déclarations, ainsi que les données nécessaires au traitement de signaux de commande d'une installation ou d'un processus. Un programme utilisateur est affecté à un module programmable (CPU ou FM, par exemple) et peut être structuré en petites entités : blocs.

Structure de programme utilisateur

La structure du programme utilisateur décrit la hiérarchie d'appel des blocs au sein d'un programme CPU et donne une vue d'ensemble des blocs utilisés ainsi que des relations de dépendance existantes.

Projet

Classeur contenant tous les objets d'une solution d'automatisation, quel que soit le nombre de stations, de modules et leur mise en réseau.

R

Relais de masquage

Le relais de masquage MCR (Master Control Relay) est utilisé pour activer ou désactiver le flux d'énergie (trajet du courant) dans les schémas à contacts de relais. Un trajet de courant désactivé correspond à une suite d'opérations qui écrit une valeur nulle à la place de la valeur calculée ou à une suite d'opérations qui conserve la valeur existant en mémoire.

Réseau

Les réseaux permettent d'organiser les blocs CONT en trajets de courant fermés.

Résultat logique

Le résultat logique (RLG) est l'état de signal actuel dans le processeur, utilisé pour le traitement de signal binaire ultérieur. L'exécution de certaines opérations dépend du RLG précédent.

S

Saisie incrémentale

Lors de la saisie incrémentale d'un bloc, le logiciel contrôle immédiatement chaque ligne ou chaque élément saisi (en ce qui concerne la syntaxe, par exemple). Les erreurs éventuelles sont signalées ; vous devez les corriger avant la fin de la saisie. La saisie incrémentale est, par exemple, possible dans les langages de programmation LIST, CONT et LOG.

Schéma à contacts (CONT)

Le schéma à contacts est un langage de programmation graphique pour STEP 5 et STEP 7 dont la représentation, conforme à la norme DIN EN-61131-3 (norme CEI 1131-3), correspond à un schéma à relais. Contrairement à ce qui se passe pour la liste d'instructions, seul un jeu d'opérations limité est représentable en CONT.

Section de déclaration de variables

La section de déclaration de variables permet de déclarer les données locales d'un bloc de code lorsqu'un programme est créé à l'aide d'un éditeur de texte.

Séquence combinatoire

Une séquence combinatoire est la partie du programme utilisateur qui commence par un bit de première interrogation (PI) dont l'état de signal est 0 et qui se termine quand une opération ou un événement remet le bit /PI à 0. Lorsque la CPU a exécuté la première opération d'une séquence combinatoire, l'état de signal du bit /PI est toujours 1. Certaines opérations comme les opérations de sortie (S, R ou =, par exemple) remettent le bit /PI à 0. *Voir aussi* Première interrogation.

Source

Une source (fichier de texte) est une partie d'un programme, créée à l'aide d'un éditeur de texte ou de graphique. A partir d'elle est généré, après compilation, le programme utilisateur exécutable ou le code machine pour M7.

Une source S7 est enregistrée sous le programme S7 dans le classeur « Sources ».

Station

Appareil pouvant être connecté comme unité à part entière à un ou plusieurs sous-réseaux (automate programmable, console de programmation, station opérateur, par exemple).

T

Table de déclaration de variables

Les données locales d'un bloc de code sont déclarées dans la table de déclaration de variables lorsque le programme est créé en mode incrémental (saisie incrémentale).

Table des mnémoniques

Table destinée à l'affectation de mnémoniques (ou noms symboliques) à des adresses pour les données globales et les blocs. Exemple : Arret_Urg (mnémonique) - E 1.7 (adresse) ou Regul (mnémonique) - SFB 24 (bloc).

Table des variables

Dans la table des variables figurent les variables, indications de format comprises, devant être visualisées et forcées.

Tableau

Un tableau (ARRAY) est un type de données complexe constitué d'éléments de données de type identique. Ces éléments de données sont soit élémentaires, soit complexes.

Trajet du courant

Caractéristiques du langage de programmation « schéma à contacts ». Dans le trajet du courant sont représentés les contacts et bobines. Il est également possible d'y intégrer des éléments complexes, tels que des fonctions mathématiques par exemple, représentés sous forme de pavés. Un trajet du courant est relié à une barre d'alimentation.

Types de données

Les types de données permettent de déterminer la manière dont le programme utilisateur utilise la valeur d'une variable ou d'une constante.

Dans SIMATIC S7, vous disposez de deux types de données conformément à la norme CEI 1131-3 :

- les types de données élémentaires,
- les types de données complexes.

Types de données complexes

Les types de données complexes sont créés par l'utilisateur à l'aide de la déclaration de types de données. Ils ne portent pas de noms qui leur sont propres et, par conséquent, ne peuvent être utilisés qu'une seule fois. On distingue les tableaux et les structures. Les types de données STRING et DATE_AND_TIME comptent également parmi les types de données complexes.

Types de données élémentaires

Les types de données élémentaires sont des types de données prédéfinis conformément à la norme CEI 1131-3.

Exemple :

- Le type de données BOOL définit une variable binaire (bit).
- Le type de données INT définit une variable entière de 16 bits.

Types de données utilisateur (UDT)

Il s'agit de structures de données spéciales que vous générez et utilisez en fonction de leur définition dans l'ensemble du programme CPU. Les types de données utilisateur peuvent soit être utilisés comme types de données élémentaires ou complexes dans la déclaration des variables de blocs de code (FC, FB, OB), soit servir de modèle pour la création de blocs de données de structure identique.

Z

Zone de mémoire

Une unité centrale SIMATIC S7 est composée de trois zones de mémoire :

- la mémoire de chargement
- la mémoire de travail et
- la mémoire système.

Index

Symboles

—( )—. *Voir* Sortie
—(#)—. *Voir* Connecteur
—(AUF). *Voir* Ouvrir bloc de données
  Voir aussi —(OPN).
—(CALL). *Voir* Appeler FC/SFC sans paramètre
—(CD), abréviation internationale. *Voir* Décrémenter
—(CU), abréviation internationale. *Voir* Incrémenter
—(JMP). *Voir* Saut si 1
—(JMPN). *Voir* Saut si 0
—(MCR<). *Voir* Relais de masquage en fonction
—(MCR>). *Voir* Relais de masquage hors fonction
—(MCRA). *Voir* Activer relais de masquage
—(MCRD). *Voir* Désactiver relais de masquage
—(N). *Voir* Détecter front descendant du RLG
—(OPN), abréviation internationale. *Voir* Ouvrir bloc de données
—(P). *Voir* Détecter front montant du RLG
—(R). *Voir* Mettre à 0
—(RET), 20-7
—(S). *Voir* Mettre à 1
—(SA). *Voir* Temporisation sous forme de retard à la retombée
—(SAVE). *Voir* Sauvegarder RLG dans RB
—(SC), abréviation internationale. *Voir* Initialiser compteur
—(SD), abréviation internationale. *Voir* Temporisation sous forme de retard à la montée
—(SE). *Voir* Temporisation sous forme de retard à la montée
—(SE), abréviation internationale. *Voir* Temporisation sous forme d'impulsion prolongée
—(SF), abréviation internationale. *Voir* Temporisation sous forme de retard à la retombée
—(SI). *Voir* Temporisation sous forme d'impulsion
—(SP), abréviation internationale. *Voir* Temporisation sous forme d'impulsion
—(SS). *Voir* Temporisation sous forme de retard à la montée mémorisé
—(SV). *Voir* Temporisation sous forme d'impulsion prolongée
—(SZ). *Voir* Initialiser compteur
—(ZR). *Voir* Décrémenter
—(ZV). *Voir* Incrémenter
—| |—. *Voir* Contacts, contact à fermeture
—|/|—. *Voir* Contacts, contact à ouverture
—|BIE|—. *Voir* Bit d'anomalie « Registre RB »
—|NOT|—. *Voir* Inverser RLG
—(OPN). *Voir* Ouvrir bloc de données
—|OS|—. *Voir* Bit d'anomalie « Débordement mémorisé »
—|OV|—. *Voir* Bit d'anomalie « Débordement »
—|RB|— , abréviation internationale. *Voir* Bit d'anomalie « Registre RB »
—|UO|—. *Voir* Bit d'anomalie « Opération illicite »
/PI. *Voir* Première interrogation

A

Accumulateurs
  description, 6-12
  fonction, 6-12
  valeur de comptage, 10-2
ACOS. *Voir* Arc cosinus
Activer relais de masquage —(MCRA), 20-9
ADD_DI. *Voir* Additionner entiers de 32 bits
ADD_I. *Voir* Additionner entiers de 16 bits
ADD_R. *Voir* Additionner nombres réels
Additionner entiers de 16 bits (ADD_I), 11-2
Additionner entiers de 32 bits (ADD_DI), 11-3
Additionner nombres réels (ADD_R), 12-3
Adressage
  définition, 7-2
  plages, 6-5
  symbolique, 3-24
Adresse, 3-7
  édition en CONT, 3-23
  symbolique, 3-24
Affectation de blocs de données d'instance
  aux FB, 4-4
  aux UDT, 4-4
Affecter valeur (MOVE), 14-2
Appel de blocs, 20-2

Appel de blocs fonctionnels (FB)
effet sur les bits du mot d'état, 20-4
paramètres, 20-6
sous forme de pavé, 20-4
Appel de blocs fonctionnels système (SFB)
effet sur les bits du mot d'état, 20-4
paramètres, 20-6
sous forme de pavé, 20-4
Appel de fonctions (FC)
avec l'opération « Appeler FC/SFC sans paramètre », 20-2
effet sur les bits du mot d'état, 20-4
paramètres, 20-6
sous forme de pavé, 20-4
Appel de fonctions système (SFC)
avec l'opération « Appeler FC/SFC sans paramètre », 20-2
effet sur les bits du mot d'état, 20-4
paramètres, 20-6
sous forme de pavé, 20-4
Appeler FC/SFC sans paramètre —(CALL), 20-2
Arc cosinus (ACOS), 12-13, 12-14
Arc sinus (ASIN), 12-13, 12-14
Arc tangente (ATAN), 12-13, 12-15
Arithmétique sur nombres à virgule flottante. *Voir* Opérations arithmétiques sur nombres réels
Arithmétique sur nombres entiers. *Voir* Opérations arithmétiques sur nombres entiers
Arithmétique sur nombres réels. *Voir* Opérations arithmétiques sur nombres réels
ARRAY. *Voir* Tableau
Arrondir à entier de 32 bits (ROUND), 14-15
ASIN. *Voir* Arc sinus
ATAN. *Voir* Arc tangente
Attributs de bloc, 5-3

B

Bascule mise à 0, mise à 1 (RS), 8-24
Bascule mise à 1, mise à 0 (SR), 8-23
Base de temps
résolution, 9-3
S5TIME, 9-2, C-9
BCD_DI. *Voir* Convertir nombre DCB en entier de 32 bits
BCD_I. *Voir* Convertir nombre DCB en entier de 16 bits
BCDF. *Voir* Erreur de conversion DCB
BI1 et BI0. *Voir* Bits indicateurs (BI1 et BI0)
Bit d'anomalie « Débordement » —|OV|—, 19-7
Bit d'anomalie « Débordement mémorisé » —|OS|—, 19-9
Bit d'anomalie « Opération illicite » —|UO|—, 19-6
par rapport aux opérations arithmétiques sur nombres réels, 19-6
Bit d'anomalie « Registre RB » —|BIE|—, 19-3
Bit d'état (ETAT), 6-14
Bit de débordement (DEB), 6-14
affecté par une opération arithmétique, 11-11, 12-7
bit d'anomalie « Débordement » —|OV|—, 19-7
Bit de débordement mémorisé (DM), 6-14
affecté par une opération arithmétique, 11-11, 12-7
bit d'anomalie « Débordement mémorisé » —|OS|—, 19-9
Bit de résultat binaire (RB), 6-16
bit d'anomalie « Registre RB » —|BIE|—, 19-3
sauvegarder RLG dans RB, 8-8
Bit OU, 6-14
Bits de résultat, opérations, 19-4
Bits du mot d'état
Voir aussi Mot d'état
modification, 6-12
Bits indicateurs (BI1 et BI0), 6-14
affectés par une opération arithmétique, 11-11, 12-7
évalués comme bits de résultat, 19-4
par rapport à « Bit d'anomalie Opération illicite », 19-6
Bloc
abandon, 20-7
appel, 20-2
chargement, 2-6
condition de déclenchement, 5-7
création, 2-5
d'organisation, 2-2
de données, 2-3
fonctionnel (FB), 2-2
ouverture, 2-5
paramétrage de l'environnement d'appel, 5-7
propriétés, 3-2
saisie en LIST, 3-13
sauvegarde, 2-5
séquence, 2-7
Bloc de code
création, 3-2
structure, 3-2
Bloc de données (DB)
associé à un UDT, 4-2
création, 4-4
global, 4-2
initialisation, 4-7
méthodes de création, 4-2, 4-4
sauvegarde des valeurs effectives, 4-7
vue des déclarations, 4-5
vue des données, 4-6
zone de mémoire, 6-4
plage d'adresses, 6-5

Bloc de données d'instance, 4-2, 20-6
création, 4-4
Bloc fonctionnel (FB)
appeler FB sous forme de pavé, 20-4
paramètres, 20-6
Bloc fonctionnel système (SFB)
appeler SFB sous forme de pavé, 20-4
paramètres, 20-6
Bloc standard, 5-3
Blocs de données (DB). *Voir* Opérations sur blocs de données
Bobine
disposition, 3-16
restrictions pour les opérations, 6-3
BOOL (bit), plage, 7-3, C-3
Branches parallèles, 3-21
création d'une nouvelle branche, 3-21
décomposition, 3-22
BYTE (octet), plage, 7-3, C-3

C

CEIL. *Voir* Convertir nombre réel en entier supérieur le plus proche
CHAR (caractère), plage, 7-3, C-3
Chargement d'une valeur de comptage, format, 10-2
Chargement d'une valeur de temps, format, 9-2
Chargement de blocs, 2-6
CMP_D. *Voir* Comparer entiers de 32 bits
CMP_I. *Voir* Comparer entiers de 16 bits
CMP_R. *Voir* Comparer nombres réels
Commentaire
dans la table de déclaration des variables, 3-7
de bloc, 3-28
de réseau, 3-28
Comparaison des états de signal, 19-4
Comparaison du résultat d'une opération arithmétique par rapport à 0, 19-4
Comparer entiers de 16 bits (CMP_I), 13-2
Comparer entiers de 32 bits (CMP_D), 13-3
Comparer nombres réels (CMP_R), 13-5
Complément à 1 d'entier de 16 bits (INV_I), 14-10
Complément à 1 d'entier de 32 bits (INV_DI), 14-11
Complément à 2 d'entier de 16 bits (NEG_I), 14-12
Complément à 2 d'entier de 32 bits (NEG_DI), 14-13
Compteur
Voir aussi Opérations de comptage
décrémental (Z_RUECK), 10-7
incrémental (Z_VORW), 10-5
Compteur
incrémental/décrémental (ZAEHLER), 10-3
valeur de comptage
format, 10-2
plage, 10-2
valeurs admissibles, 10-2
zone de mémoire, 6-4, 10-2
plage d'adresses, 6-5
Compteurs, opérations
décrémenter —(ZR), 8-13
incrémenter —(ZV), 8-12
initialiser compteur —(SZ), 8-11
Condition de déclenchement, paramétrage, 5-7
Conflit d'horodatage, 3-9
Connecteur —(#)—, 8-6
Connexion de contacts
en parallèle, 6-10–6-18
en série, 6-8
CONT, signification, 1-1
Contacts
connexion en parallèle, 6-10
connexion en série, 6-8, 6-9
contact à fermeture —| |—, 6-6, 8-3
contact à ouverture —|/|—, 6-7, 8-4
Conversion. *Voir* Opérations de conversion
Convertir entier de 16 bits en entier de 32 bits (I_DI), 14-6
Convertir entier de 16 bits en nombre DCB (I_BCD), 14-5
Convertir entier de 32 bits en nombre DCB (DI_BCD), 14-8
Convertir entier de 32 bits en nombre réel (DI_R), 14-9
Convertir nombre DCB en entier de 16 bits (BCD_I), 14-4
Convertir nombre DCB en entier de 32 bits (BCD_DI), 14-7
Convertir nombre réel en entier inférieur le plus proche (FLOOR), 14-18
Convertir nombre réel en entier supérieur le plus proche (CEIL), 14-17
COS. *Voir* Cosinus
Cosinus (COS), 12-13, 12-14
Couleur des sélections, 3-18
Court-circuit, 3-15
CPU, registres, 6-12
Création d'un DB, choix de la méthode, 4-4
Création de programmes utilisateur, 3-2

D

DATE (date CEI), plage, 7-3, C-3

Date et heure, type de données.
Voir DATE_AND_TIME
DATE_AND_TIME (date et heure), description, C-9
DCB. *Voir* Nombre décimal codé binaire
DEB. *Voir* Bit de débordement (DEB)
Décalage. *Voir* Opérations de décalage
Décalage vers droite d'entier de 16 bits (SHR_I), 16-7
Décalage vers droite d'entier de 32 bits (SHR_DI), 16-9
Décalage vers droite de double mot (SHR_DW), 16-6
Décalage vers droite de mot (SHR_W), 16-5
Décalage vers gauche de double mot (SHL_DW), 16-4
Décalage vers gauche de mot (SHL_W), 16-2
Décrémenter —(ZR), 8-13
Désactiver relais de masquage —(MCRD), 20-9
Détecter front descendant de signal (NEG), 8-22
Détecter front descendant du RLG —(N), 8-20
Détecter front montant de signal (POS), 8-21
Détecter front montant du RLG —(P), 8-19
DI_BCD. *Voir* Convertir entier de 32 bits en nombre DCB
DI_R. *Voir* Convertir entier de 32 bits en nombre réel
DINT (entier de 32 bits)
description, C-4
plage, 7-3, C-3
DIV_DI. *Voir* Diviser entiers de 32 bits
DIV_I. *Voir* Diviser entiers de 16 bits
DIV_R. *Voir* Diviser nombres réels
Diviser entiers de 16 bits (DIV_I), 11-8
Diviser entiers de 32 bits (DIV_DI), 11-9
Diviser nombres réels (DIV_R), 12-6
DM. *Voir* Bit de débordement mémorisé (DM)
Données locales, zone de mémoire, 6-4
plage d'adresses, 6-5
Durée SIMATIC, type de données. *Voir* S5TIME
DWORD (double mot), plage, 7-3, C-3

E

Editeur, paramètres pour CONT, 3-3
Editeur CONT
incrémental, 2-4
lancement, 2-4
Edition
pendant l'exécution du programme, 2-7
table de déclaration des variables, 3-8
Eléments CONT, saisie, 3-19
EN. *Voir* Entrée de validation (EN)
EN / ENO, signification, 6-17
ENO. *Voir* Sortie de validation (ENO)
Entrée de validation (EN), paramètres, 6-3
Environnement d'appel. *Voir* Condition de déclenchement
Environnement de test
laboratoire, 5-8
processus, 5-8
sélection, 5-8
Erreur de conversion DCB, 14-4, 14-7
ET
connexion de contacts en série, 6-8
double mot (WAND_DW), 15-4
mot (WAND_W), 15-3
table de vérité, 6-9
ETAT. *Voir* Bit d'état (ETAT)
Etat. *Voir* Opérations sur bits d'état
Etat de programme, 5-5
Voir aussi Test
lancement/arrêt, 5-8
Etat de programme CONT, paramétrage, 5-6
Etat de signal
détecter front descendant, 8-22
détecter front montant, 8-21
Exemples
applications pratiques des opérations, B-2
opérations arithmétiques sur nombres entiers, B-13
opérations combinatoires sur bits, B-3
opérations combinatoires sur mots, B-14
opérations de comptage et de comparaison, B-11
opérations de temporisation, B-7

F

Famille pour la sélection d'éléments CONT, 3-20
FLOOR. *Voir* Convertir nombre réel en entier inférieur le plus proche
Flux d'énergie, 6-6
Fonctions (FC), 2-2
  appeler FC sous forme de pavé, 20-4
  appeler FC/SFC sans paramètre, 20-2
  paramètres, 20-6
Fonctions système (SFC)
  appeler FC/SFC sans paramètre, 20-2
  appeler SFC sous forme de pavé, 20-4
  paramètres, 20-6
Fonctions trigonométriques d'angles, 12-13
Format
  valeur de comptage, 10-2
  valeur de temps, 9-2
Format des nombres, C-2
  DATE_AND_TIME (date et heure), C-9
  décimal codé binaire (DCB), C-8
  DINT (entier de 32 bits), C-4
  INT (entier de 16 bits), C-3
  REAL (réel), C-5
  S5TIME (durée SIMATIC), C-9
Front descendant de signal. *Voir* Détecter front descendant de signal
Front descendant du RLG. *Voir* Détecter front descendant du RLG
Front montant de signal. *Voir* Détecter front montant de signal
Front montant du RLG. *Voir* Détecter front montant du RLG

G

Gestion d'exécution de programme. *Voir* Opérations de gestion d'exécution de programme

I

I_BCD. *Voir* Convertir entier de 16 bits en nombre DCB
I_DI. *Voir* Convertir entier de 16 bits en entier de 32 bits
Imbrication, relais de masquage (MCR), 20-13
Incrémenter —(ZV), 8-12
Influence du mot d'état
  EN = 0, 6-17
  EN = 1, 6-17
Information mnémoniques, 3-24
Initialisation, 4-7
Initialiser compteur —(SZ), 8-11
INT (entier de 16 bits)
  description, C-3
  plage, 7-3, C-3
Interrogation des bits indicateurs (BI1 et BI0), 6-14
INV_DI. *Voir* Complément à 1 d'entier de 32 bits
INV_I. *Voir* Complément à 1 d'entier de 16 bits
Inverser le signe d'un nombre réel (NEG_R), 14-14
Inverser RLG —|NOT|—, 8-7

J

Jonction, décomposition, 3-27

L

LABEL. *Voir* Repère de saut
Langage de programmation, définition, 2-5
Largeur des colonnes, dans la table de déclaration des variables, 3-7
Liste alphabétique des opérations
  désignations françaises avec abréviations SIMATIC correspondantes, A-2
  désignations françaises avec désignations internationales correspondantes, A-5
  désignations internationales avec abréviations correspondantes, A-9
  désignations internationales avec désignations françaises correspondantes, A-12
Logique booléenne, 6-6

M

Mémento, zone de mémoire, 6-4
  plage d'adresses, 6-5
Mémoire image des entrées (MIE), zone de mémoire, 6-4
  plage d'adresses, 6-5
Mémoire image des sorties (MIS), zone de mémoire, 6-4
  plage d'adresses, 6-5
Méthode de création d'un DB, choix, 4-4
Mettre à 0 —(R), 8-10
Mettre à 1 —(S), 8-9
Mnémoniques, 3-24
MOD. *Voir* Reste de division (32 bits)
Mode de substitution, 3-26
Mode de test, sélection, 5-8

Modes de représentation. *Voir* Format des nombres
Mot d'état
bit d'anomalie « Débordement » —|OV|—, 19-7
bit d'anomalie « Débordement mémorisé » —|OS|—, 19-9
bit d'anomalie « Registre RB », 19-3
bit d'état (ETAT), 6-14
bit de débordement (DEB), 6-14
bit de débordement mémorisé (DM), 6-14
bit de résultat binaire (RB), 6-16
bit OU, 6-14
bits affectés par le résultat d'une opération arithmétique, 12-7
bits indicateurs (BI1 et BI0), 6-14
évalués comme bits de résultat, 19-4
description, 6-12
effet de l'appel d'une FC, d'une SFC, d'un FB ou d'un SFB, 20-4
modification des bits, 6-12
opérations sur bits d'état, 19-2
organisation, 6-12
première interrogation (/PI), 6-13
résultat logique (RLG), 6-13
structure, 19-2
MOVE. *Voir* Affecter valeur
MUL_DI. *Voir* Multiplier entiers de 32 bits
MUL_I. *Voir* Multiplier entiers de 16 bits
MUL_R. *Voir* Multiplier nombres réels
Multi-instances
appel, 3-10, 3-20
déclaration, 3-10
règles, 3-10
Multiplier entiers de 16 bits (MUL_I), 11-6
Multiplier entiers de 32 bits (MUL_DI), 11-7
Multiplier nombres réels (MUL_R), 12-5

N

NEG. *Voir* Détecter front descendant de signal
NEG_DI. *Voir* Complément à 2 d'entier de 32 bits
NEG_I. *Voir* Complément à 2 d'entier de 16 bits
NEG_R. *Voir* Inverser le signe d'un nombre réel
Nombre décimal codé binaire, C-8
Nombres à virgule flottante, type de données. *Voir* REAL
Nombres entiers
Voir aussi Opérations arithmétiques sur nombres entiers
comparer entiers de 16 bits, 13-2
comparer entiers de 32 bits, 13-3
Nombres entiers de 16 bits, type de données. *Voir* INT
Nombres entiers de 32 bits, type de données. *Voir* DINT
Nombres réels
Voir aussi Opérations arithmétiques sur nombres réels
comparer nombres réels, 13-5
Noms symboliques. *Voir* Mnémoniques

O

Opérande
description, 7-4
élément avec opérande et valeur, 6-2
éléments possibles, 7-2
repère de saut pour opération de saut, 18-2
types, 7-4
Opération OU en CONT, 3-21
Opérations
applications pratiques, B-2
dépendant du relais de masquage (MCR), 20-8
liste alphabétique
désignations françaises avec abréviations SIMATIC correspondantes, A-2
désignations françaises avec désignations internationales correspondantes, A-5
désignations internationales avec abréviations correspondantes, A-9
désignations internationales avec désignations françaises correspondantes, A-12
opérations combinatoires sur bits, 8-2
sous forme d'éléments avec opérande, 6-2
sous forme d'éléments avec opérande et valeur, 6-2
sous forme de pavés avec paramètres, 6-3
Opérations arithmétiques sur nombres entiers
additionner entiers de 16 bits (ADD_I), 11-2
additionner entiers de 32 bits (ADD_DI), 11-3
application pratique, B-13
diviser entiers de 16 bits (DIV_I), 11-8
diviser entiers de 32 bits (DIV_DI), 11-9
multiplier entiers de 16 bits (MUL_I), 11-6
multiplier entiers de 32 bits (MUL_DI), 11-7
reste de division, 32 bits (MOD), 11-10
soustraire entiers de 16 bits (SUB_I), 11-4
soustraire entiers de 32 bits (SUB_DI), 11-5
Opérations arithmétiques sur nombres réels, 12-2
additionner nombres réels (ADD_R), 12-3
affectant les bits du mot d'état, 12-7
arc cosinus (ACOS), 12-13, 12-14
arc sinus (ASIN), 12-13, 12-14
arc tangente (ATAN), 12-13, 12-15
cosinus (COS), 12-13, 12-14

diviser nombres réels (DIV_R), 12-6
multiplier nombres réels (MUL_R), 12-5
par rapport à « Bit d'anomalie Opération illicite », 19-6
plage autorisée des résultats, 11-11, 12-7
sinus (SIN), 12-13
soustraire nombres réels (SUB_R), 12-4
tangente (TAN), 12-13, 12-15
Opérations combinatoires sur bits, 8-2
application pratique, B-3–B-6
bascule mise à 0, mise à 1 (RS), 8-24
bascule mise à 1, mise à 0 (SR), 8-23
Connecteur —(#)—, 8-6
contact à fermeture —| |—, 8-3
contact à ouverture —|/|—, 8-4
décrémenter —(ZR), 8-13
détecter front descendant de signal (NEG), 8-22
détecter front descendant du RLG —(N), 8-20
détecter front montant de signal (POS), 8-21
détecter front montant du RLG —(P), 8-19
incrémenter —(ZV), 8-12
initialiser compteur —(SZ), 8-11
inverser RLG —|NOT|—, 8-7
mettre à 0 —(R), 8-10
mettre à 1 —(S), 8-9
sauvegarder RLG dans RB —(SAVE), 8-8
sortie —( )—, 8-5
temporisation sous forme d'impulsion —(SI), 8-14
temporisation sous forme d'impulsion prolongée —(SV), 8-15
temporisation sous forme de retard à la montée —(SE), 8-16
temporisation sous forme de retard à la montée mémorisé —(SS), 8-17
temporisation sous forme de retard à la retombée —(SA), 8-18
Opérations combinatoires sur mots, 15-2
application pratique, B-14–B-16
ET double mot (WAND_DW), 15-4
ET mot (WAND_W), 15-3
OU double mot (WOR_DW), 15-6
OU exclusif double mot (WXOR_DW), 15-8
OU exclusif mot (WXOR_W), 15-7
OU mot (WOR_W), 15-5
Opérations de comparaison
application pratique, B-11
comparer entiers de 16 bits (CMP_I), 13-2
comparer entiers de 32 bits (CMP_D), 13-3
comparer nombres réels (CMP_R), 13-5
Opérations de comptage
application pratique, B-11
compteur décrémental (Z_RUECK), 10-7
compteur incrémental (Z_VORW), 10-5
compteur incrémental/décrémental (ZAEHLER), 10-3
Opérations de conversion
arrondir à entier de 32 bits (ROUND), 14-15
complément à 1 d'entier de 16 bits (INV_I), 14-10
complément à 1 d'entier de 32 bits (INV_DI), 14-11
complément à 2 d'entier de 16 bits (NEG_I), 14-12
complément à 2 d'entier de 32 bits (NEG_DI), 14-13
convertir entier de 16 bits en entier de 32 bits (I_DI), 14-6
convertir entier de 16 bits en nombre DCB (I_BCD), 14-5
convertir entier de 32 bits en nombre DCB (DI_BCD), 14-8
convertir entier de 32 bits en nombre réel (DI_R), 14-9
convertir nombre DCB en entier de 16 bits (BCD_I), 14-4
convertir nombre DCB en entier de 32 bits (BCD_DI), 14-7
convertir nombre réel en entier inférieur le plus proche (FLOOR), 14-18
convertir nombre réel en entier supérieur le plus proche (CEIL), 14-17
inverser le signe d'un nombre réel (NEG_R), 14-14
tronquer à la partie entière (32 bits) (TRUNC), 14-16
Opérations de décalage, 16-2
décalage vers droite d'entier de 16 bits (SHR_I), 16-7
décalage vers droite d'entier de 32 bits (SHR_DI), 16-9
décalage vers droite de double mot (SHR_DW), 16-6
décalage vers droite de mot (SHR_W), 16-5
décalage vers gauche de double mot (SHL_DW), 16-4
décalage vers gauche de mot (SHL_W), 16-2
Opérations de gestion d'exécution de programme
activer relais de masquage —(MCRA), 20-9
appeler FB, FC, SFB, SFC sous forme de pavé, 20-4
appeler FC/SFC sans paramètre —(CALL), 20-2
désactiver relais de masquage —(MCRD), 20-9
relais de masquage en fonction —(MCR<), 20-12
relais de masquage hors fonction —(MCR>), 20-12
retour —(RET), 20-7

Opérations de rotation, 16-10
rotation vers droite de double mot (ROR_DW), 16-11
rotation vers gauche de double mot (ROL_DW), 16-10
Opérations de saut, 18-2
repère de saut (LABEL), 18-6
repère de saut comme opérande, 18-2
saut si 0 —(JMPN), 18-5
saut si 1 —(JMP), 18-3, 18-4
Opérations de temporisation, 9-2
application pratique, B-7
temporisation sous forme d'impulsion (S_IMPULS), 9-5
temporisation sous forme d'impulsion prolongée (S_VIMP), 9-7
temporisation sous forme de retard à la montée (S_EVERZ), 9-9
temporisation sous forme de retard à la montée mémorisé (S_SEVERZ), 9-11
temporisation sous forme de retard à la retombée (S_AVERZ), 9-13
Opérations de transfert, affecter valeur (MOVE), 14-2
Opérations du relais de masquage, 20-8
Opérations sur bits d'état, 19-2
bit d'anomalie « Débordement » —|OV|—, 19-7
bit d'anomalie « Débordement mémorisé » —|OS|—, 19-9
bit d'anomalie « Opération illicite » —|UO|—, 19-6
bit d'anomalie « Registre RB » —|BIE|—, 19-3
bits de résultat, 19-4
Opérations sur blocs de données, ouvrir bloc de données —(AUF), 17-2
OU
Voir aussi Bit OU
connexion de contacts en parallèle, 6-10
connexion de contacts en parallèle avec « Contact à fermeture », 6-11–6-18
création de branches parallèles, 3-21
table de vérité, 6-11
OU double mot (WOR_DW), 15-6
OU exclusif double mot (WXOR_DW), 15-8
OU exclusif mot (WXOR_W), 15-7
complément à 1 d'entier de 16 bits, 14-10
complément à 1 d'entier de 32 bits, 14-11
OU mot (WOR_W), 15-5
Ouverture d'un bloc, 2-5
Ouvrir bloc de données —(AUF), 17-2

P

Paramètres
entrée de validation (EN), 6-3
entrées et sorties d'un pavé, 6-3
pour éditeur CONT, 3-3, 3-23
pour état de programme CONT, 5-6
sortie de validation (ENO), 6-3
Pavé
disposition, 3-16
opération, 6-2
restrictions concernant les opérations, 6-3
Périphérie d'entrée : entrées externes, zone de mémoire, 6-4
plage d'adresses, 6-5
Périphérie de sortie : sorties externes, zone de mémoire, 6-4
plage d'adresses, 6-5
Plages d'adresses, zones de mémoire, 6-5
POS. *Voir* Détecter front montant de signal
Première interrogation (/PI), 6-13
résultat logique, 6-13
Programmation, applications pratiques, B-2
Programmation symbolique, exemple, B-3, B-4, B-5
Programme utilisateur
création, 2-4, 3-2
structure, 2-2
Propriétés de bloc, 3-2
traitement, 5-2
Propriétés de blocs de données, 4-2
Protection du bloc, 5-3
Protection en écriture, 5-3
Protection Know_How, 5-3

R

RB. *Voir* Bit de résultat binaire (RB)
REAL (réel)
champs des composants, C-6
description, C-4
exemples, C-7
format, C-5
plage, 7-3, C-3
Recherche d'erreurs, dans le bloc, 3-23
Registres de la CPU, 6-12
fonction, 6-12
mot d'état, 6-12
valeur dans la cellule de temporisation, 9-3
valeur de comptage, 10-2

Règles pour la saisie des éléments CONT, 3-15
Relais de masquage (MCR), 20-8
activer relais de masquage —(MCRA), 20-9
désactiver relais de masquage —(MCRD), 20-9
imbrication des opérations, 20-13
réactions des opérations « Mise à 1 » —(S) et « Mise à 0 » —(R), 20-8
relais de masquage en fonction —(MCR<), 20-12
relais de masquage hors fonction —(MCR>), 20-12
Relais de masquage en fonction —(MCR<), 20-12
Relais de masquage hors fonction —(MCR>), 20-12
Repère de saut, comme opérande d'opération de saut, 18-2
Repère de saut (LABEL), 18-6
Représentation
des mnémoniques, 3-24
des nombres. *Voir* Format des nombres
Réseau
fermeture en CONT, 3-15
insertion, 3-18
sélection, 3-18
Résolution des valeurs de temps. *Voir* Base de temps
Reste de division, 32 bits (MOD), 11-10
Restrictions pour les pavés et les bobines, 6-3
Résultat binaire. *Voir* Bit de résultat binaire
Résultat logique (RLG), 6-6, 6-9, 6-13
description, 6-6
détecter front descendant, 8-20
détecter front montant, 8-19
inversion, 8-7
première interrogation (/PI), 6-13
Résultat logique d'une combinaison. *Voir* Résultat logique (RLG)
Retour —(RET), 20-7
RLG. *Voir* Résultat logique (RLG)
ROL_DW. *Voir* Rotation vers gauche de double mot
ROR_DW. *Voir* Rotation vers droite de double mot
Rotation. *Voir* Opérations de rotation
Rotation vers droite de double mot (ROR_DW), 16-11
Rotation vers gauche de double mot (ROL_DW), 16-10
ROUND. *Voir* Arrondir à entier de 32 bits
RS. *Voir* Bascule mise à 0, mise à 1

S

S_AVERZ. *Voir* Temporisation sous forme de retard à la retombée
S_CD, abréviation internationale. *Voir* Compteur décrémental
S_CU, abréviation internationale. *Voir* Compteur incrémental
S_CUD, abréviation internationale. *Voir* Compteur incrémental/décrémental
S_EVERZ. *Voir* Temporisation sous forme de retard à la montée
S_IMPULS. *Voir* Temporisation sous forme d'impulsion
S_ODT, abréviation internationale. *Voir* Temporisation sous forme de retard à la montée
S_ODTS, abréviation internationale. *Voir* Temporisation sous forme de retard à la montée mémorisé
S_OFFDT, abréviation internationale. *Voir* Temporisation sous forme de retard à la retombée
S_PEXT, abréviation internationale. *Voir* Temporisation sous forme d'impulsion prolongée
S_PULSE, abréviation internationale. *Voir* Temporisation sous forme d'impulsion
S_SEVERZ. *Voir* Temporisation sous forme de retard à la montée mémorisé
S_VIMP. *Voir* Temporisation sous forme d'impulsion prolongée
S5TIME
base de temps, 9-2, C-9
description, C-9
plage, 7-3, C-3
valeur de temps, 9-2
Saisie d'éléments CONT, 3-18, 3-19
Saut. *Voir* Opérations de saut
Saut conditionnel, 18-4
Saut inconditionnel, 18-3
Saut si 0 —(JMPN), 18-5
Saut si 1 —(JMP), 18-3, 18-4
Sauvegarde d'un bloc, 2-5
Sauvegarder RLG dans RB —(SAVE), 8-8
Schéma à contacts, 1-1
Section d'instructions, 3-2
constitution, 3-13
dans CONT, 3-4
traitement, 3-13
Sélection dans les réseaux, 3-18
Sens du courant, 3-15
Séquence d'opérations combinatoires
définition, 6-13
première opération, 6-13
Séquence des blocs, 2-7
SFB, 2-4
SFC, 2-4
SHL_DW. *Voir* Décalage vers gauche de double mot
SHL_W. *Voir* Décalage vers gauche de mot
SHR_DI. *Voir* Décalage vers droite d'entier de 32 bits

SHR_DW. *Voir* Décalage vers droite de double mot
SHR_I. *Voir* Décalage vers droite d'entier de 16 bits
SHR_W. *Voir* Décalage vers droite de mot
SIMATIC Manager, 2-4
SIN. *Voir* Sinus
Sinus (SIN), 12-13
Sortie —( ), 8-5
Sortie de validation (ENO)
Voir aussi Bit de résultat binaire (RB)
paramètres, 6-3
Soustraire entiers de 16 bits (SUB_I), 11-4
Soustraire entiers de 32 bits (SUB_DI), 11-5
Soustraire nombres réels (SUB_R), 12-4
SR. *Voir* Bascule mise à 1, mise à 0
Structure, saisie dans la table de déclaration des variables, 3-8
SUB_DI. *Voir* Soustraire entiers de 32 bits
SUB_I. *Voir* Soustraire entiers de 16 bits
SUB_R. *Voir* Soustraire nombres réels
Substitution
de jonctions en CONT, 3-27
des adresses/paramètres, 3-26
des éléments CONT, 3-26
Syntaxe, vérification, 4-5

T

Table de déclaration
pour blocs de données, 4-2
structure lors de la création du DB, 4-5
traitement, 4-5
Table de déclaration des variables, 3-2, 3-4, 3-6
édition, 3-8
objet, 3-6
structure, 3-6
Table de vérité
ET, 6-9
OU, 6-11
Table des mnémoniques, 3-24
Tableau, saisie dans la table de déclaration des variables, 3-9
TAN. *Voir* Tangente
Tangente (TAN), 12-13, 12-15
Temporisation
Voir aussi Opérations de temporisation
composants, 9-2
lecture de la valeur et de la base de temps, 9-3
valeur de temps, 9-2
plage, 9-2
syntaxe, 9-2
valeurs admissibles, 9-2
Temporisation
vue d'ensemble des types de temporisation, 9-4
zone de mémoire, 6-4, 9-2
plage d'adresses, 6-5
Temporisation sous forme d'impulsion (S_IMPULS), 9-5
Temporisation sous forme d'impulsion —(SI), 8-14
Temporisation sous forme d'impulsion prolongée (S_VIMP), 9-7
Temporisation sous forme d'impulsion prolongée —(SV), 8-15
Temporisation sous forme de retard à la montée (S_EVERZ), 9-9
Temporisation sous forme de retard à la montée —(SE), 8-16
Temporisation sous forme de retard à la montée mémorisé (S_SEVERZ), 9-11
Temporisation sous forme de retard à la montée mémorisé —(SS), 8-17
Temporisation sous forme de retard à la retombée (S_AVERZ), 9-13
Temporisation sous forme de retard à la retombée —(S), 8-18
Temporisations, opérations
temporisation sous forme d'impulsion —(SI), 8-14
temporisation sous forme d'impulsion prolongée —(SV), 8-15
temporisation sous forme de retard à la montée —(SE), 8-16
temporisation sous forme de retard à la montée mémorisé —(SS), 8-17
temporisation sous forme de retard à la retombée —(SA), 8-18
Test
dans l'éditeur CONT, 5-5
des programmes CONT, 5-5
TIME (durée CEI), plage, 7-3, C-3
TIME_OF_DAY (heure du jour), plage, 7-3, C-3
Titre
de bloc, 3-28
de réseau, 3-28
Traitement de la table de déclaration, 4-5
Transfert. *Voir* Opérations de transfert
Tronquer à la partie entière (32 bits) (TRUNC), 14-16
TRUNC. *Voir* Tronquer à la partie entière (32 bits)
Type de déclaration
modification, 3-8
signification, 3-7
Type de données, dans la table de déclaration des variables, 3-7

Type de données utilisateur (UDT)
application, 4-8
création, 4-8
définition, 4-2
Type de donnnées utilisateur (UDT), 2-4
Types d'opérandes, 7-4
Types de données, 7-3, C-2
BOOL (bit), 7-3, C-3
BYTE (octet), 7-3, C-3
CHAR (caractère), 7-3, C-3
DATE (date CEI), 7-3, C-3
DATE_AND_TIME (date et heure), C-9
définis par l'utilisateur, 2-4
DINT (entier de 32 bits), 7-3, C-3, C-4
DWORD (double mot), 7-3, C-3
INT (entier de 16 bits), 7-3, C-3
modes de représentation, C-2
REAL (réel), 7-3, C-3, C-4
S5TIME (durée SIMATIC), 7-3, C-3, C-9
TIME (durée CEI), 7-3, C-3
TIME_OF_DAY (heure du jour), 7-3, C-3
WORD (mot), 7-3, C-3

U

UDT. *Voir* Type de données utilisateur

V

Valeur de comptage
format, 10-2
plage, 10-2
Valeur de temps, 9-3
format dans la cellule de temporisation, 9-3
lecture, 9-3
plage, 9-3
syntaxe, 9-2
Valeur effective, initialisation, 4-7
Valeur en cours, dans la vue des données, 4-6
Valeur initiale, 3-7
pour blocs de données, 4-6
Variable, 3-7
Vérification
de la syntaxe, 4-5
du temps de cycle, 5-9
Vue des déclarations, 4-5
Vue des données, 4-6

W

WAND_DW. *Voir* ET double mot
WAND_W. *Voir* ET mot
WOR_DW. *Voir* OU double mot
WOR_W. *Voir* OU mot
WORD (mot), plage, 7-3, C-3
WXOR_DW. *Voir* OU exclusif double mot
WXOR_W. *Voir* OU exclusif mot

Z

Z_RUECK. *Voir* Compteur décrémental
Z_VORW. *Voir* Compteur incrémental
ZAEHLER. *Voir* Compteur
incrémental/décrémental
Zones de mémoire, 6-4
blocs de données, 6-4
compteurs, 6-4
données locales, 6-4
mémentos, 6-4
mémoire image des entrées, 6-4
mémoire image des sorties, 6-4
périphérie d'entrée : entrées externes, 6-4
périphérie de sortie : sorties externes, 6-4
plages d'adresses, 6-5
temporisations, 6-4

Siemens AG
AUT E 146
Östliche Rheinbrückenstr. 50
76187 Karlsruhe
République Féderale d'Allemagne

Expéditeur :
Vos Nom : _
Fonction : _
Entreprise : _
Rue : _
Ville : _
Téléphone : _

Indiquez votre secteur industriel :

- ❒ Industrie automobile
- ❒ Industrie chimique
- ❒ Industrie électrique
- ❒ Industrie du alimentaire
- ❒ Contrôle/commande
- ❒ Construction mécanique
- ❒ Pétrochimie
- ❒ Industrie pharmaceutique
- ❒ Traitement des matières plastiques
- ❒ Industrie du papier
- ❒ Industrie textile
- ❒ Transports
- ❒ Autres _ _ _ _ _ _ _ _ _ _ _

Remarques/suggestions

Vos remarques et suggestions nous permettent d'améliorer la qualité générale de notre documentation. C'est pourquoi nous vous serions reconnaissants de compléter et de renvoyer ces formulaires à Siemens.

Répondez aux questions suivantes en donnant votre évaluation comprise entre 1 pour très bien et 5 pour très mauvais.

1. Le contenu du manuel répond-il à votre attente ? ☐
2. Les informations requises peuvent-elles facilement être trouvées ? ☐
3. Le texte est-il compréhensible ? ☐
4. Le niveau des détails techniques répond-il à votre attente ? ☐
5. Quelle évaluation attribuez-vous aux figures et tableaux ? ☐

Les ligne suivantes vous permettent d'exposer des problèmes concrets que vous auriez éventuellement rencontrés :

_ _ _ _